FPGA ARCHITECTURE OF ADAPTIVE FILTER USING MULTIPRECISION MULTIPLIER

PROJECT REPORT

Submitted by

MOHANRAJ.P
Reg. no. 13MAE09

in partial fulfillment for the requirement of award of the degree

of

MASTER OF ENGINEERING

in

APPLIED ELECTRONICS

Department of Electronics and Communication Engineering

KUMARAGURU COLLEGE OF TECHNOLOGY
(An Autonomous Institution affiliated to Anna University, Chennai)
COIMBATORE - 641049

ANNA UNIVERSITY: CHENNAI 600 025

April 2015

BONAFIDE CERTIFICATE

Certified that this project report titled **"FPGA ARCHITECTURE OF ADAPTIVE FILTER USING MULTIPRECISION MULTIPLIER"** is the bonafide work of **P.MOHANRAJ [Reg. No. 13MAE09]** who carried out the project under my supervision. Certified further that, to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Ms. K.THILAGAVATHI	**Dr. RAJESWARI MARIAPPAN**
PROJECT SUPERVISOR	**HEAD OF THE DEPARTMENT**
Department of ECE	Department of ECE
Kumaraguru College of Technology	Kumaraguru College of Technology
Coimbatore-641 049	Coimbatore-641 049

The Candidate with **Register No. 13MAE09** was examined by us in the project viva –voice examination held on...........................

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

First I would like to express my praise and gratitude to the Lord, who has showered his grace and blessing enabling me to complete this project in an excellent manner. He has made all things in beautiful in his time.

I express my sincere thanks to our beloved Joint Correspondent, **Shri. Shankar Vanavarayar** for his kind support and for providing necessary facilities to carry out the project work.

I would like to express my sincere thanks to our beloved Principal **Dr.R.S.Kumar M.E., Ph.D.,** who encouraged us with his valuable thoughts.

I would like to express my sincere thanks and deep sense of gratitude to our HOD, **Dr. Rajeswari Mariappan M.E., Ph.D.,** for her valuable suggestions and encouragement which paved way for the successful completion of the project.

I am greatly privileged to express my deep sense of gratitude to the Project Coordinator **Ms.S.Sasikala M.Tech**, Associate Professor, for her continuous support throughout the course.

In particular, I wish to thank and express my everlasting gratitude to the Supervisor **Ms.K.Thilagavathi M.E.**, Assistant Professor (SRG) for her expert counseling in each and every steps of project work.

I would also convey my gratitude to **Mr.S.Govidaraju M.E.,** Professor**,** Department of ECE, for his immense contribution and valuable advice throughout the project and I wish to convey my deep sense of gratitude to all teaching and non-teaching staff members of ECE Department for their help and cooperation.

Finally I thank my parents and my family members for giving me the moral support and abundant blessings in all of my activities and my dear friends who helped me to endure my difficult times with their unfailing support and warm wishes.

ABSTRACT

Adaptive filters, as a part of digital signal systems, has been broadly used in many applications such as adaptive noise cancellation, adaptive beam forming, system identification and channel equalization. The proposed work is to design FPGA architecture of an adaptive LMS filter with low power. A Multiprecision (MP) reconfigurable multiplier that incorporates variable precision, parallel processing (PP), razor-based dynamic voltage scaling (DVS) and dedicated MP operands scheduling to provide optimum performance for a variety of operating conditions is proposed. Combining Multiprecision (MP) with dynamic voltage scaling (DVS) can provide a dramatic reduction in power consumption by adjusting the supply voltage, frequency according to circuit run-time workload rather than fixed. Razor based flip flop technique is used to correct the timing errors. To improve the speed of the operation in partial product generation during multiplication process the proposed method is implemented using modified Booth algorithm. Power consumption of adaptive filter design using Multiprecision Multiplier is also compared with adaptive filter design using array multiplier as well as adaptive filter design using Modified Booth algorithm without Multiprecision Multiplier.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

CMOS	Complementary Metal Oxide Semiconductor
CSA	Carry Save Adder
DRAM	Dynamic Random Access Memory
DSP	Digital Signal Processing
DVFM	Dynamic Voltage and Frequency Management
DVS	Dynamic Voltage Scaling
EEG	Electro Encepho Gram
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
IOS	Input Operands Scheduler
ISI	Inter Symbol Interference
LMS	Least Mean Square
LRLS	Lattice Recursive Least Square
LTI	Linear Time Invariant
MAC	Multiplier And Accumulator
MBA	Modified Booth Algorithm
MP	Multi Precision
MPEG	Moving Picture Experts Group
NLMS	Normalized Least Mean Square
PE	Processing Element
RLS	Recursive Least Square
SNR	Signal to Noise power Ratio

VCO	Voltage Controlled Oscillator
VFMU	Voltage / Frequency Management Unit
VHDL	Very high speed integrated circuit Hardware Description Language
VSU	Voltage Scaling Unit

CHAPTER I

INTRODUCTION

1.1 ADAPTIVE FILTER

An adaptive filter is a computational device that attempts to model the relationship between two signals in real time in an iterative manner. Adaptive filters are often realized either as a set of program instructions running on an arithmetical processing device such as a microprocessor or DSP chip, or as a set of logic operations implemented in a field-programmable gate array(FPGA) or in a semicustom or custom VLSI integrated circuit. However, ignoring any errors introduced by numerical precision effects in these implementations, the fundamental operation of an adaptive filter can be characterized independently of the specific physical realization that it takes. For this reason, we shall focus on the mathematical forms of adaptive filters as opposed to their specific realizations in software or hardware. Descriptions of adaptive filters as implemented on DSP chips and on a dedicated integrated circuit.

An adaptive filter is defined by four aspects:

∗ The signals being processed by the filter
∗ The structure that defines how the output signal of the filter is computed from its input signal
∗ The parameters within this structure that can be iteratively changed to alter the filter's input-output relationship
∗ The adaptive algorithm that describes how the parameters are adjusted from one time instant to the next

By choosing a particular adaptive filter structure, one specifies the number and type of parameters that can be adjusted. The adaptive algorithm used to update the parameter values of the system can take on a myriad of forms and is often derived as a form of optimization procedure that minimizes an error criterion that is useful for the task at hand.

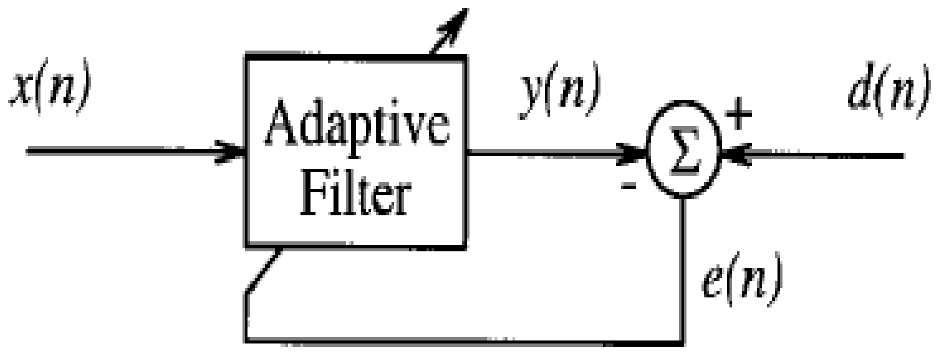


Fig 1.1 General Adaptive filter

Where x (n) is the input signal to a linear filter block at time n

y (n) is the corresponding output signal

d (n) is the desired input signal to the adaptive filter

e (n) is the error signal that denotes the difference between desired signal d (n) and output signal y (n)

Figure 1.1 shows a block diagram in which a sample from a digital input signal x(n) is fed into a device, called an adaptive filter, that computes a corresponding output signal sample y(n) at time n. For the moment, the structure of the adaptive filter is not important, except for the fact that it contains adjustable parameters whose values affect how y (n) is computed. The output signal is compared to a second signal d (n), called the desired response signal, by subtracting the two samples at time n. This difference signal, given by

$$e(n) = d(n) - y(n)$$

is known as the error signal. The error signal is fed into a procedure which alters or adapts the parameters of the filter from time n to time (n+1) in a well-defined manner. This process of adaptation is represented by the oblique arrow that pierces the adaptive filter block in the figure. As the time index *n* is incremented, it is hoped that the output of the adaptive filter becomes a better and better match to the desired response signal through this adaptation process, such that the magnitude of e (n) decreases over time. In the adaptive filtering task, adaptation refers to the method by which the parameters of the system are changed from time index n to time index (n+1). The number and types of parameters within this system depend on the computational structure chosen for the system.

1.2 APPLICATIONS OF ADAPTIVE FILTERS

Adaptive filters are used in many applications. They are described in details.

1.2.1 SYSTEM IDENTIFICATION

Consider Fig. 1.2, which shows the general problem of system identification. In this diagram, the system enclosed by dashed lines is a “black box,” meaning that the quantities inside are not observable from the outside. Inside this box is (1) an unknown system which represents a general input output relationship and (2) the signal η(n), called the observation noise signal because it corrupts the observations of the signal at the output of the unknown system.

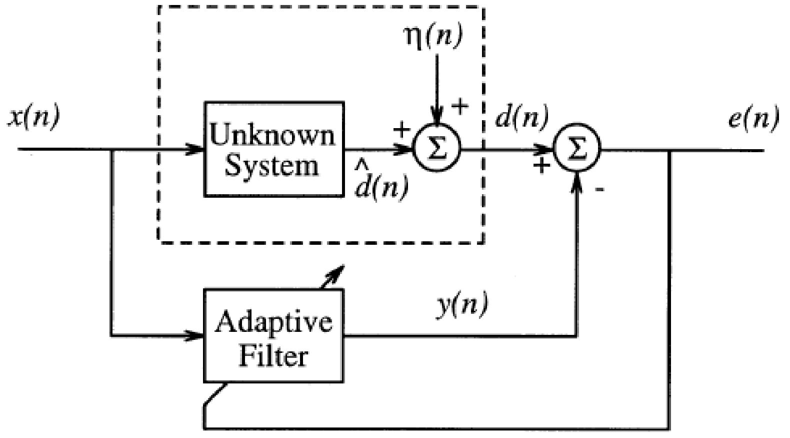


Fig 1.2 System Identification

Let d ^(n) represent the output of the unknown system with x (n) as its input. Then, the desired response signal in this model is

d (n) = d ^(n) + η(n)

Here, the task of the adaptive filter is to accurately represent the signal d ^(n) at its output. If y (n) = d ^(n), then the adaptive filter has accurately modeled or identified the portion of the unknown system that is driven by x (n).

Since the model typically chosen for the adaptive filter is a linear filter, the practical goal of the adaptive filter is to determine the best linear model that describes the input-output relationship of the unknown system. Such a procedure makes the most sense when the unknown system is also a linear model of the same structure as the adaptive filter, as it is possible that y(n) = d^(n), for some set of adaptive filter parameters. For ease of discussion, let the unknown system and the adaptive filter both be FIR filters, such that

$$d(n) = W_{opt}^{T}(n)\, X(n) + \eta(n)$$

Where W_{opt} (n) is an optimum set of filter coefficients for the unknown system at time *n*. In this problem formulation, the ideal adaptation procedure would adjust w (n) such that W (n) = Wopt (n) as n→∞. In practice, the adaptive filter can only adjust W (n) such that y (n) closely approximates d ^(n) over time.

The system identification task is at the heart of numerous adaptive filtering applications. We list several of these applications here.

Channel Identification

In communication systems, useful information is transmitted from one point to another across a medium such as an electrical wire, an optical fiber, or a wireless radio link. Non idealities of the transmission medium or channel distort the fidelity of the transmitted signals, making the deciphering of the received information difficult. In cases where the effects of the distortion can be modeled as a linear filter, the resulting “smearing” of the transmitted symbols is known as inter-symbol interference (ISI). In such cases, an adaptive filter can be used to model the effects of the channel ISI for purposes of deciphering the received information in an optimal manner. In this problem scenario, the transmitter sends to the receiver a sample sequence x (n) that is known to both the transmitter and receiver. The receiver then attempts to model the received signal d (n) using an adaptive filter whose input is the known transmitted sequence x (n). After a suitable period of adaptation, the parameters of the adaptive filter in W (n) are fixed and then used in a procedure to decode future signals transmitted across the channel. Channel identification is typically employed when the fidelity of the transmitted channel is severely compromised or when simpler techniques for sequence detection cannot be used.

Plant Identification

In many control tasks, knowledge of the transfer function of a linear plant is required by the physical controller so that a suitable control signal can be calculated and applied. In such cases, we can characterize the transfer function of the plant by exciting it with a known signal x (n) and then attempting to match the output of the plant d (n) with a linear adaptive filter. After a suitable period of adaptation, the system has been adequately modeled, and the resulting adaptive filter coefficients in W (n) can be used in a control scheme to enable the overall closed-loop system to behave in the desired manner. In certain scenarios, continuous updates of the plant transfer function estimate provided by W (n) are needed to allow the controller to function properly.

Echo Cancellation for long-Distance Transmission

In voice communication across telephone networks, the existence of junction boxes called hybrids near either end of the network link hampers the ability of the system to cleanly transmit voice signals. Each hybrid allows voices that are transmitted via separate lines or channels across a long-distance network to be carried locally on a single telephone line, thus lowering the wiring costs of the local network. However, when small impedance mismatches between the long distance lines and the hybrid junctions occur, these hybrids can reflect the transmitted signals back to their sources, and the long transmission times of the long-distance network—about 0*:*3 s for a trans-oceanic call via a satellite link—turn these reflections into a noticeable echo that makes the understanding of conversation difficult for both callers. The traditional solution to this problem prior to the advent of the adaptive filtering solution was to introduce significant loss into the long-distance network so that echoes would decay to an acceptable level before they became perceptible to the callers. Unfortunately, this solution also reduces the transmission quality of the telephone link and makes the task of connecting long distance calls more difficult.

An adaptive filter can be used to cancel the echoes caused by the hybrids in this situation. Adaptive filters are employed at each of the two hybrids within the network. The input x(n) to each adaptive filter is the speech signal being received prior to the hybrid junction, and the desired response signal d(n) is the signal being sent out from the hybrid across the long-distance connection. The adaptive filter attempts to model the transmission characteristics of the hybrid junction as well as any echoes that appear across the long-distance portion of the network. When the system is properly designed, the error signal e (n) consists almost totally of the local talker's speech signal, which is then transmitted over the network. Such systems were first proposed in the mid-1960s and are commonly used today.

Acoustic Echo Cancellation

A related problem to echo cancellation for telephone transmission systems is that of acoustic echo cancellation for conference-style speakerphones. When using a speakerphone, a caller would like to turn up the amplifier gains of both the microphone and the audio loudspeaker in order to transmit and hear the voice signals more clearly. However, the feedback path from the device's loudspeaker to its input microphone causes a distinctive howling sound if these gains are too high. In this case, the culprit is the room's response to the voice signal being broadcast by the speaker; in effect, the room acts as an extremely poor hybrid junction, in analogy with the echo cancellation task discussed previously. A simple solution to this problem is to only allow one person to speak at a time, a form of operation called half-duplex transmission. However, studies have indicated that half-duplex transmission causes problems with normal conversations, as people typically overlap their phrases with others when conversing.

To maintain full-duplex transmission, an acoustic echo canceller is employed in the speakerphone to model the acoustic transmission path from the speaker to the microphone. The input signal x (n) to the acoustic echo canceller is the signal being sent to the speaker, and the desired response signal d (n) is measured at the microphone on the device. Adaptation of the system occurs continually throughout a telephone call to model any physical changes in the room acoustics. Such devices are readily available in the marketplace today. In addition, similar technology can and is used to remove the echo that occurs through the combined radio/room/telephone transmission path when one places a call to a radio or television talk show.

Adaptive Noise Cancelling

When collecting measurements of certain signals or processes, physical constraints often limit our ability to cleanly measure the quantities of interest. Typically, a signal of interest is linearly mixed with other extraneous noises in the measurement process, and these extraneous noises introduce unacceptable errors in the measurements. However, if a linearly related reference version of any one of the extraneous noises can be cleanly sensed at some other physical location in the system, an adaptive filter can be used to determine the relationship between the noise reference x(n) and the component of this noise that is contained in the measured signal d(n). After adaptively subtracting out this component, what remains in e (n) is the signal of interest. If several extraneous noises corrupt the measurement of interest, several adaptive filters can be used in parallel as long as suitable noise reference signals are available within the system.

Adaptive noise cancelling has been used for several applications. One of the first was a medical application that enabled the electroencephalogram (EEG) of the fetal heartbeat of an unborn child to be cleanly extracted from the much-stronger interfering EEG of the maternal heartbeat signal.

1.2.2 INVERSE MODELING

We now consider the general problem of inverse modeling, as shown in Fig 1.3. In this diagram, a source signal s (n) is fed into an unknown system that produces the input signal x (n) for the adaptive filter. The output of the adaptive filter is subtracted from a desired response signal that is a delayed version of the source signal, such that

$$d(n) = s(n-\Delta)$$

where Δ is a positive integer value. The goal of the adaptive filter is to adjust its characteristics such that the output signal is an accurate representation of the delayed source signal.

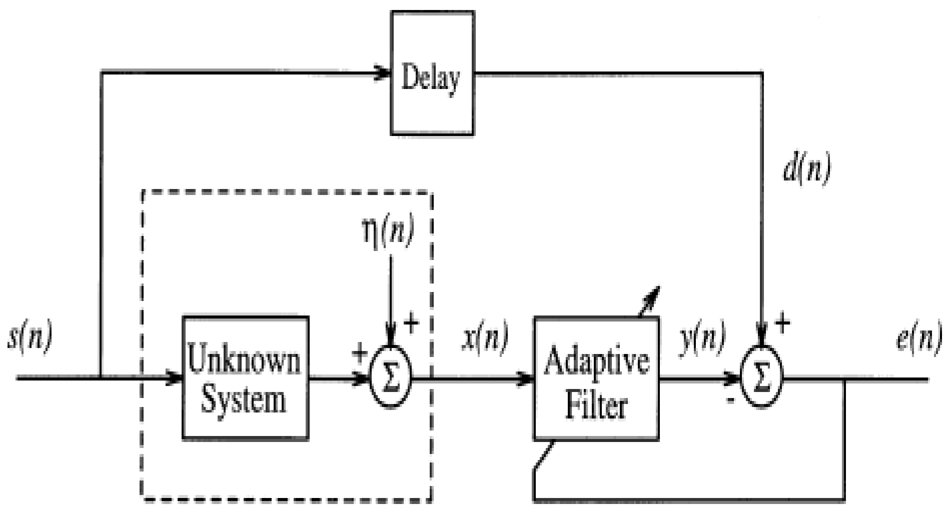


Fig 1.3 Inverse Modeling

The inverse modeling task characterizes several adaptive filtering applications, two of which are now described.

Channel Equalization

Channel equalization is an alternative to the technique of channel identification described previously for the decoding of transmitted signals across non ideal communication channels. In both cases, the transmitter sends a sequence s (n) that is known to both the transmitter and receiver. However, in equalization, the received signal is used as the input signal x (n) to an adaptive filter, which adjusts its characteristics so that its output closely matches a delayed version s (n-Δ) of the known transmitted signal. After a suitable adaptation period, the coefficients of the system either are fixed and used to decode future transmitted messages or are adapted using a crude estimate of the desired response signal that is computed from y (n). This latter mode of operation is known as decision-directed adaptation.

Channel equalization was one of the first applications of adaptive filters and is described in the pioneering work of Lucky. Today, it remains as one of the most popular uses of an adaptive filter. Practically every computer telephone modem transmitting at rates of 9600 baud (bits per second) or greater contains an adaptive equalizer. Adaptive equalization is also useful for wireless communication systems. Qureshi provides a tutorial on adaptive equalization. A related problem to equalization is deconvolution, a problem that appears in the context of geophysical exploration. Equalization is closely related to linear prediction, a topic that we shall discuss shortly.

Inverse Plant Modeling

In many control tasks, the frequency and phase characteristics of the plant hamper the convergence behavior and stability of the control system. We can use a system of the form in Fig 1.3 to compensate for the nonideal characteristics of the plant and as a method for adaptive control. In this case, the signal s (n) is sent at the output of the controller, and the signal x (n) is the signal measured at the output of the plant. The coefficients of the adaptive filter are then adjusted so that the cascade of the plant and adaptive filter can be nearly represented by the pure delay $z^{-\Delta}$.

1.2.3 LINEAR PREDICTION

A third type of adaptive filtering task is shown in Fig 1.4. In this system, the input signal x (n) is derived from the desired response signal as

$$x(n) = d(n-\Delta)$$

where Δ is an integer value of delay. In effect, the input signal serves as the desired response signal, and for this reason it is always available. In such cases, the linear adaptive filter attempts to predict future values of the input signal using past samples, giving rise to the name linear prediction for this task.

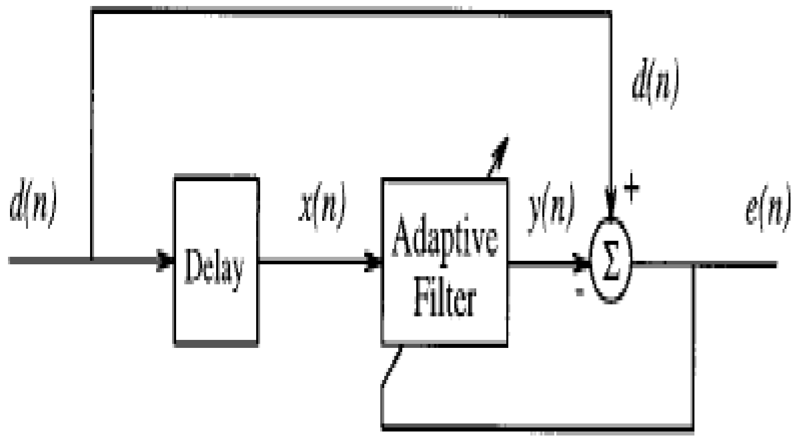


Fig 1.4 Linear Prediction

If an estimate of the signal x (n + Δ) at time n is desired, a copy of the adaptive filter whose input is the current sample x (n) can be employed to compute this quantity. However, linear prediction has a number of uses besides the obvious application of forecasting future events, as described in the following two applications.

Linear Predictive Coding

When transmitting digitized versions of real-world signals such as speech or images, the temporal correlation of the signals is a form of redundancy that can be exploited to code the waveform in a smaller number of bits than are needed for its original representation. In these cases, a linear predictor can be used to model the signal correlations for a short block of data in such a way as to reduce the number of bits needed to represent the signal waveform. Then, essential information about the signal model is transmitted along with the coefficients of the adaptive filter for the given data block. Once received, the signal is synthesized using the filter coefficients and the additional signal information provided for the given block of data.

When applied to speech signals, this method of signal encoding enables the transmission of understandable speech at only 2.4 kb/s, although the reconstructed speech has a distinctly synthetic quality. Predictive coding can be combined with a quantizer to enable higher-quality speech encoding at higher data rates using an adaptive differential pulse-code modulation

(ADPCM) scheme. In both of these methods, the lattice filter structure plays an important role because of the way in which it parameterizes the physical nature of the vocal tract.

Adaptive Line Enhancement

In some situations, the desired response signal d(n) consists of a sum of a broadband signal and a nearly periodic signal, and it is desired to separate these two signals without specific knowledge about the signals (such as the fundamental frequency of the periodic component).

In these situations, an adaptive filter configured as in Fig 1.4 can be used. For this application, the delay Δ is chosen to be large enough such that the broadband component in x (n) is uncorrelated with the broadband component in x (n- Δ) In this case, the broadband signal cannot be removed by the adaptive filter through its operation, and it remains in the error signal e (n) after a suitable period of adaptation. The adaptive filter's output y (n) converges to the narrowband component, which is easily predicted given past samples. The name line enhancement arises because periodic signals are characterized by lines in their frequency spectra, and these spectral lines are enhanced at the output of the adaptive filter.

1.2.4 FEED FORWARD CONTROL

Another problem area combines elements of both the inverse modeling and system identification tasks and typifies the types of problems encountered in the area of adaptive control known as feed forward control. Fig 1.5 shows the block diagram for this system, in which the output of the adaptive filter passes through a plant before it is subtracted from the desired response to form the error signal. The plant hampers the operation of the adaptive filter by changing the amplitude and phase characteristics of the adaptive filter's output signal as represented in e (n). Thus, knowledge of the plant is generally required in order to adapt the parameters of the filter properly.

An application that fits this particular problem formulation is active noise control, in which unwanted sound energy propagates in air or a fluid into a physical region in space. In such cases, an electro acoustic system employing microphones, speakers, and one or more adaptive filters can be used to create a secondary sound field that interferes with the unwanted sound, reducing its level in the region via destructive interference. Similar techniques can be used to reduce vibrations in solid media.

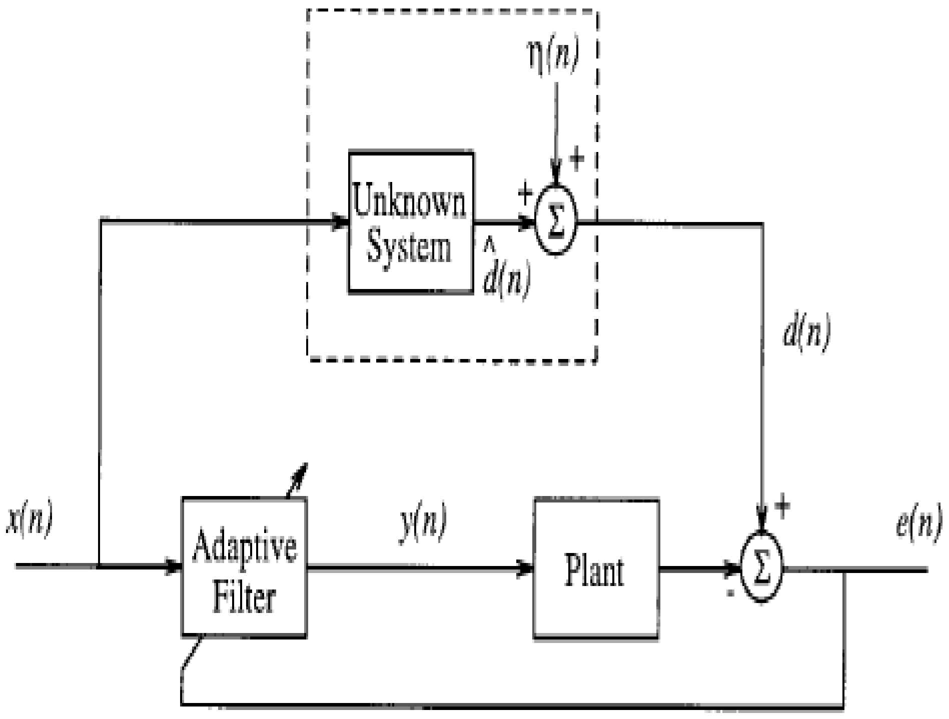


Fig 1.5 Feed Forward Control

CHAPTER II

LITERATURE SURVEY

Hesam Ariyadoost, Yousef S. Kavian, and Karim Ansari-As [7], “Performance Evaluation of LMS and DLMS Digital Adaptive FIR Filters by Realization on FPGA,” proposes to implement the adaptive digital Least Mean Square (LMS) and delayed-LMS (DLMS) Finite Impulse Response (FIR) filters on Field Programmable Gate Array (FPGA) chips for typical noise cancellation applications and compare the behavior of LMS and DLMS adaptive algorithms in terms of chip area utilization and the filter critical path time or filter frequency. The direct FIR architecture is considered for filter designing and the VHDL hardware description language is used for algorithm modeling. The obtained results by the synthesize tool QUARTUS II on a single STRATIX II chip, EP2S15F484C3, from ALTERA Inc. demonstrate that the DLMS algorithm which has a pipeline architecture is faster than LMS algorithm while it uses more chip area due to the usage of extra registers.

M. Nakai, S. Akui, et al. [15], "Dynamic voltage and frequency management for a low-power embedded microprocessor," proposes a dynamic voltage and frequency management techniques. In DVFM approach, clock frequency is autonomously and dynamically controlled while voltage is adaptively controlled at the same time. A delay synthesizer in the DVC circuit emulates and provides the circuit delay information while the DFC circuit determines optimum operating frequency for the microprocessor to perform desired functions efficiently. The DVFM scheme autonomously controls clock frequency from 8 to 123 MHz in steps of 0.5 MHz and also adaptively controls supply voltage from 0.9 to 1.6 V in steps of 5 mV, achieving 82% power reduction in Personal Information Management scheduler application and 40% power reduction in MPEG4 movie playback. This low-power embedded microprocessor, fabricated with 0.18- m CMOS embedded DRAM technology, enables high-performance operations such as audio and video applications. As process technology shrinks, this adaptive leakage power compensation scheme will become more important in realizing high-performance and low-power mobile consumer applications.

S. Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull and D. T. Blaauw [16], “Razor II: In Situ Error Detection and Correction for PVT and SER Tolerance”, proposes the traditional methods of adaptive design which uses look-up tables or so-called “canary” circuits. In the look-up table based approach, the design is pre-characterized to obtain voltage and frequency pairs for which correct operation is guaranteed. This approach exploits periods of low CPU utilization by dynamically scaling voltage and frequency, thereby obtaining energy savings. However, each operating point must be suitably margined to guarantee computational correctness in the worst-case combination of process, voltage and temperature (PVT) conditions. The canary-circuit based approach eliminates a subset of these worst-case margins by using a delay-chain which mimics the critical path of the actual design. The propagation delay through this replica-path is monitored and voltage and frequency are scaled until the replica-path just about fails to meet timing. The replica-path tracks the critical-path delay across inter-die process variations and global fluctuations in supply voltage and temperature, thereby eliminating margins due to global PVT variations. However, the replica-path does not share the same ambient environment as the critical-path, because its on-die location differs.

Seok-Jae Lee, Student Member, IEEE, Ji-Woong Choi, Senior Member, IEEE, Seon Wook Kim, Member, IEEE, and Jongsun Park, Member, IEEE [17], “A Reconfigurable FIR Filter Architecture to Trade Off Filter Performance for Dynamic Power Consumption” proposes a simple yet efficient low power reconfigurable FIR filter architecture, where the filter order can be dynamically changed depending on the amplitude of both the filter coefficients and the inputs. In other words, when the data sample multiplied to the coefficient is so small as to mitigate the effect of partial sum in FIR filter, the multiplication operation can be simply cancelled. The filter performance degradation can be minimized by controlling the error bound as small as the quantization error or signal to noise power ratio (SNR) of given system. The primary goal of this work is to reduce the dynamic power of the FIR filter, and the main contributions are summarized as follows. 1) A new reconfigurable FIR filter architecture with real-time input and coefficient monitoring circuits is presented. Since the basic filter structure is not changed, it is applicable to the FIR filter with programmable coefficients or adaptive filters. 2) The mathematical analysis of the power saving and filter performance degradation on the proposed approach is provided. The analysis is verified using experimental results, and it can be used as a guideline to design low power reconfigurable filters.

Shweta S. Khobragade, Swapnali P. Karmore [18], "Review on: Low Power VLSI Design of Modified Booth Multiplier" proposes Low power VLSI circuits which became very vital criteria for designing the energy efficient electronic designs for prime performance and compact devices. Multipliers play a very important role for planning energy economical processors that decides the potency of the processor. To scale back the facility consumption of multiplier factor booth coding methodology is being employed to rearrange the input bits. The operation of the booth decoder is to rearrange the given booth equivalent. Booth decoder can increase the range of zeros in variety. Hence the switching activity are going to be reduced that further reduces the power consumption of the design. The input bit constant determines the switching activity part that's once the input constant is zero, the corresponding rows or column of the multiplier ought to be deactivated. When multiplicand contains more number of zeros the higher power reduction can takes place. So in modified booth multiplier high power reductions will be achieved

M. Bhardwaj, R. Min and A. Chandrakasan [19], "Quantifying and Enhancing Power-Awareness of VLSI Systems" proposes an increasingly important figure-of-merit of a VLSI system which is "power awareness," its ability to scale power consumption in response to changing operating conditions. These changes might be brought about by the time-varying nature of inputs, desired output quality, or just environmental conditions. Regardless of whether they were engineered for being power aware, systems display variations in power consumption as conditions change. This implies, by the definition above, that all systems are naturally power aware to some extent. However, one would expect that some systems are "more" power aware than others. Equivalently, we should be able to re-engineer systems to increase their power awareness. In this paper, we attempt to quantitatively define power awareness and how such awareness can be enhanced using a systematic technique.

Chong, Kwen-Siong, Bah-Hwee Gwee, Chang, Joseph.S [20], "A micro power low-voltage multiplier with reduced spurious switching" proposes a micro power 16 16-bit multiplier

(18.8 W/MHz @1.1 V) for low-voltage power-critical low speed (5 MHz) applications including hearing aids. The micro power operation is obtained by substantially reducing (by 62% and 79% compared to conventional 16 16-bit and 32 32-bit designs respectively) the spurious switching in the Adder Block in the multiplier. The approach use latches to synchronize the inputs to the adders in the Adder Block in a predetermined chronological sequence. The hardware penalty of the latches is small because the latches are integrated (as opposed to external latches) into the adder, termed the Latch Adder (LA). By means of the LAs and timing, the number of switching (spurious and that for computation) is reduced from 56 and 10 per adder in the Adder Block in conventional 16 16-bit and 32 32-bit designs respectively to 2 in our designs.

Stefania Perri, Pasquale Corsonello, Maria Antonia Iachino, Marco Lanuzza, and Giuseppe Cocorullo [21],"Variable precision arithmetic circuits for FPGA-based multimedia Processors" proposes a new efficient variable precision arithmetic circuits for field programmable gate array(FPGA)-based processors. The proposed circuits can adapt themselves to different data word lengths, avoiding time and power consuming reconfiguration. This is made possible by the introduction of on purpose designed auxiliary logic, which enables the new circuits to operate in single instruction multiple data (SIMD) fashion and allows high parallelism levels to be guaranteed when operations on lower precisions are executed. The new SIMD structures have been designed to optimally exploit the resources of a widely used family of SRAM-based FPGAs, but their architectures can be easily adapted to any either SRAM-based or antifuse-based FPGA chips.

Thomas D. Burd, Trevor A. Pering, Anthony J. Stratakos, Robert W. Brodersen [22], "A dynamic voltage scaled microprocessor system" In this paper, a microprocessor system is presented in which the supply voltage and clock frequency can be dynamically varied so that the system can deliver high throughput when required while significantly extending battery life during the low speed periods. The system consists of a dc-dc switching regulator, an ARM V4 microprocessor with a 16-kB cache, a bank of 64-kB SRAM ICs, and an I/O interface IC. This paper describes a new design technique that dynamically varies the supply voltage which provides high throughput when required. This technique can decrease the system's average energy consumption by up to 10x, without sacrificing perceived throughput, by exploiting the time-varying computational load that is commonly found in portable electronic devices.

The proposed work is to design a LMS adaptive filter design using Multiprecision multiplier, where multiplier is designed using modified booth algorithm.

CHAPTER III

PROPOSED METHODOLOGY

3.1 DESIGN OF MULTI PRECISION MULTIPLIER

3.1.1 OVERALL MP MULTIPLIER STRUCTURE

The proposed MP multiplier system (Fig 3.1) comprises five different modules that are as follows:

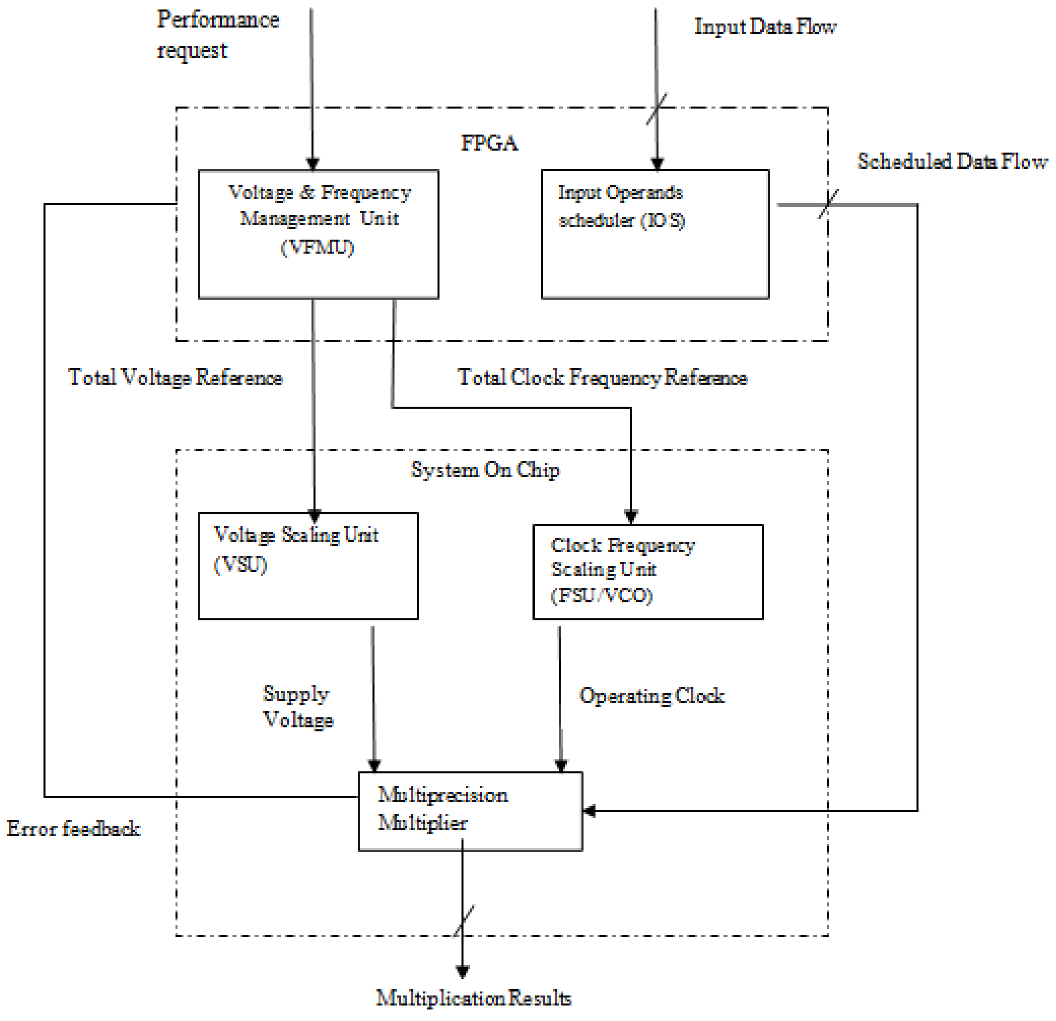

Performance request
Input Data Flow
FPGA
Scheduled Data Flow
Voltage & Frequency Management Unit (VFMU)
Input Operands scheduler (IOS)
Total Voltage Reference
Total Clock Frequency Reference
System On Chip
Voltage Scaling Unit (VSU)
Clock Frequency Scaling Unit (FSU/VCO)
Supply Voltage
Operating Clock
Multiprecision Multiplier
Error feedback
Multiplication Results

Fig 3.1 overall multiplier system architecture

∗ The MP multiplier

∗ The input operands scheduler (IOS) whose function is to reorder the input data stream into a buffer, hence to reduce the required power supply voltage transitions

∗ The frequency scaling unit implemented using a voltage controlled oscillator (VCO). Its function is to generate the required operating frequency of the multiplier

∗ The voltage scaling unit (VSU) implemented using a volt- age dithering technique to limit silicon area overhead. Its function is to dynamically generate the supply voltage so as to minimize power consumption

∗ The dynamic voltage/frequency management unit (VFMU) that receives the user requirements (e.g., throughput)

The VFMU sends control signals to the VSU and FSU to generate the required power supply voltage and clock frequency for MP multiplier.

The MP multiplier is responsible for all computations. It is equipped with razor flip-flops that can report timing errors associated to insufficiently high voltage supply levels. The operation principle is as follows. Initially, the multiplier operates at a standard supply voltage of 3.3 V. If the razor flip- flops of the multiplier do not report any errors, this means that the supply voltage can be reduced. This is achieved through the VFMU, which sends control signals to the VSU, hence to lower the supply voltage level. When the feedback provided by the razor flip-flops indicates timing errors, the scaling of the power supply is stopped.

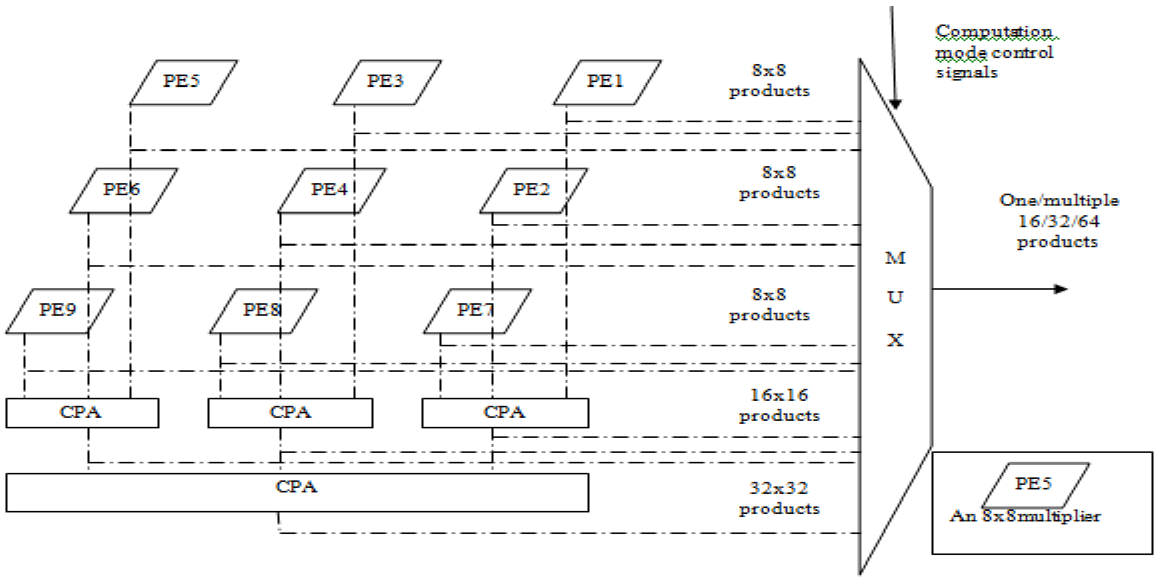


Fig 3.2 Possible configuration modes of proposed MP multiplier

The proposed multiplier (Fig 3.2) not only combines MP and DVS but also parallel processing (PP). Our multiplier comprises 8 × 8 bit reconfigurable multipliers. These building blocks can either work as nine independent multipliers or work in parallel to perform one, two or three 16 × 16 bit multiplications or a single-32 × 32 bit operation. PP can be used to increase the throughput or reduce the supply voltage level for low power operation.

3.1.2 INPUT OPERANDS SCHDULING

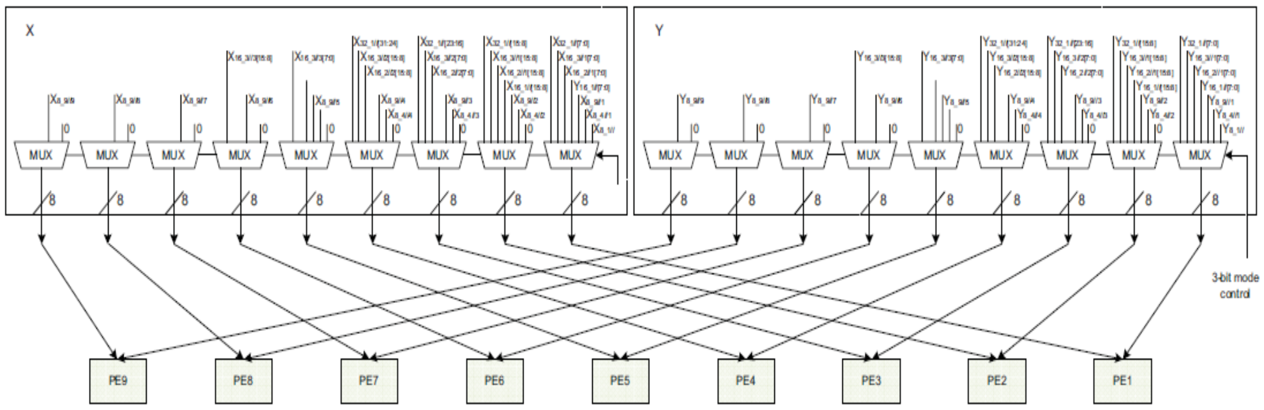


Fig 3.3 Structure of input interface unit

Fig 3.3 shows the structure of the input interface unit, which is a sub module of the MP multiplier (Fig 3.1). The role of this input interface unit (Fig. 3.3) is to distribute the input data between the nine independent processing elements (PEs) (Fig 3.2) of the 32 × 32 bit MP multiplier, considering the selected operation mode. The input interface unit uses an extra MSB sign bit to enable both signed and unsigned multiplications. A 3-bit control bus indicates whether the inputs are 1/4/9 pair(s) of 8-bit operands, or 1/2/3 pair(s) of 16-bit operands, or 1 pair of 32-bit operands, respectively. Depending on the selected operating mode, the input data stream is distributed (Fig. 3.3) between the PEs to perform the computation. A 3-bit control word defines which PEs work concurrently and which PEs are disabled. Whenever the full precision (32 × 32 bit) is not exercised, the supply voltage and the clock frequency may be scaled down according to the actual workload.

3.1.3 DYNAMIC VOLTAGE AND FREQUENCY SCALING UNIT

This section describes our dynamic voltage scaling scheme, is a power management technique. This is a method of reducing the average power consumption which is accomplished by reducing the switching losses of the system by selectively reducing the frequency and voltage of the system. Depends upon the operand scheduler it will assign the voltage and frequency for that particular operation. So the overall power consumption is reduced. DVS unit is used in a lot of applications. They are

Cell phones
MP3 players
PDAs etc;

3.1.4 RAZOR BASED FLIP FLOP TECHNIQUES

Although the worst case paths are very rarely exercised, traditional DVS approaches still maintain relatively large safety margins to ensure reliable circuit operation, resulting in excessive power dissipated. The razor technology is a breakthrough work, which largely eliminates the safety margins by achieving variable tolerance through in-situ timing error detection and correction ability.

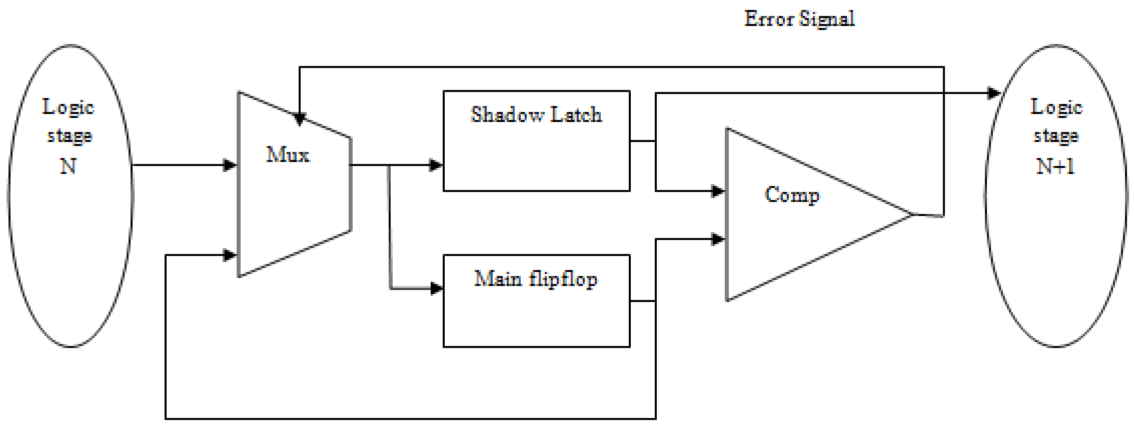


Fig 3.4 Conceptual view of razor flip-flops

This approach is based on a razor flip-flop, which detects and corrects delay errors by double sampling. The razor flip-flop (Fig 3.4) operates as a standard positive edge triggered flip flops coupled with a shadow latch, which samples at the negative edge. Therefore, the input data is given in the duration of the positive clock phase to settle down to its correct state before being sampled by the shadow latch. The minimum allowable supply voltage needs to be set; hence the shadow latch (Fig 3.4) always clocks the correct data even for the worst case conditions. This requirement is usually satisfied given that the shadow latch is clocked later than the main flip-flop. A comparator flags a timing error when it detects a discrepancy between the speculative data sampled at the main flip-flop and the correct data sampled at the shadow latch. The correct data would subsequently overwrite the incorrect signal. The key idea behind razor flip- flops is that if an error is detected at a given pipeline stage *X*, then computations are only re-executed through the following pipeline stage X + 1. This is possible because the correct sampled value would be held by the shadow latch. This approach ensures forward progress of data through the entire pipeline at the cost of a single-clock cycle.

An error correction mechanism, based on global clock gating, is implemented in the proposed multiplier. In this correction scheme, error and clock signals are used to deter- mine when the entire pipeline needs to be stalled for a single- clock cycle. Fig 3.1 shows that a global error signal is fed to the VFMU so as to alert the controlling unit whenever the current operating voltage is lower than necessary. The VFMU will then increase the voltage reference. This will in turn result in the VSU generating a new supply voltage level based on the new target voltage reference. When an error occurs, results can be recomputed at any pipeline stage using the corresponding input of the shadow latch. Therefore, the correct values can be forwarded to the corresponding next stages. Given that all stages can carry out these re-computations in parallel, the adopted global clock gating can tolerate any number of errors within a given clock cycle. After one clock cycle, normal pipeline operation can resume. The actual implementation of razor flip-flops requires careful design to meet timing constraints and avoid system failure.

3.2 BINARY ARRAY MULTIPLIER

Multipliers play an important role in today’s digital signal processing and various other applications. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets – high speed, low power consumption, regularity of layout and hence less area or even combination of them in one multiplier thus making them suitable for various high speed, low power and compact VLSI implementation. The common multiplication method is “add and shift” algorithm. In parallel multipliers number of partial products to be added is the main parameter that determines the performance of the multiplier. To reduce the number of partial products to be added, Modified Booth algorithm is one of the most popular algorithms. To achieve speed improvements Wallace Tree algorithm can be used to reduce the number of sequential adding stages. Further by combining both Modified Booth algorithm and Wallace Tree technique we can see advantage of both algorithms in one multiplier. However with increasing parallelism, the amount of shifts between the partial products and intermediate sums to be added will increase which may result in reduced speed, increase in silicon area due to irregularity of structure and also increased power consumption due to increase in interconnect resulting from complex routing. On the other hand “serial-parallel” multipliers compromise speed to achieve better performance for area and power consumption. The selection of a parallel or serial multiplier actually depends on the nature of application. In this lecture we introduce the multiplication algorithms and architecture and compare them in terms of speed, area, power and combination of these metrics.

Array multiplier is well known due to its regular structure. Multiplier circuit is based on add and shift algorithm. Each partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added. The addition can be performed with normal carry propagate adder. N-1 adders are required where N is the multiplier length.

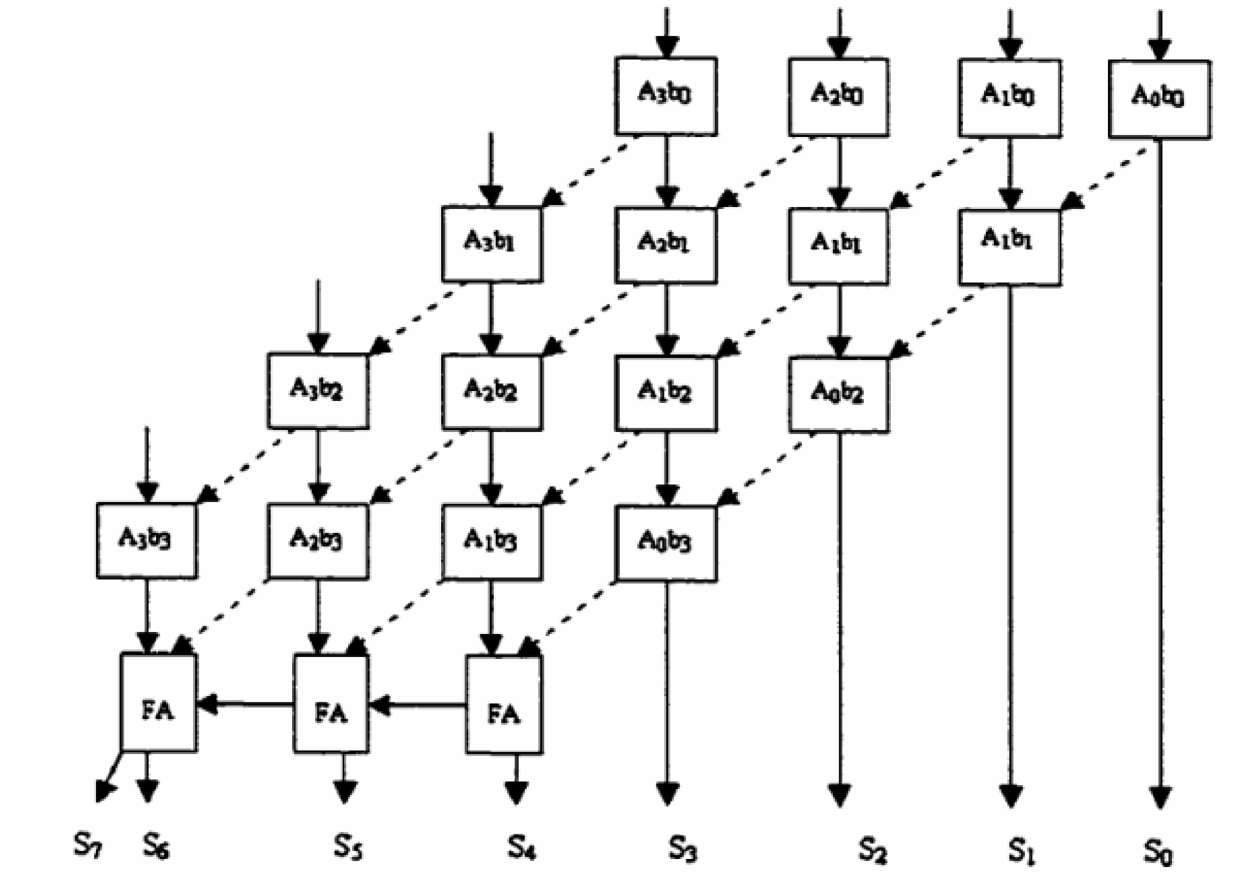


Fig 3.5 Binary Array Multiplier

Each of the ANDed terms is referred to as a partial product. The final product (the result) is formed by accumulating (summing) down each column of partial products. Any carries must be propagated from the right to the left across the columns.

Since we are dealing with binary numbers, the partial products reduce to simple AND operations between the corresponding bits in the multiplier and multiplicand. The sums down each column can be implemented using one or more 1-bit binary adders. Any adder that may need to accept a carry from the right must be a full adder. If there is no possibility of a carry propagating in from the right, then a half adder can be used instead, if desired (a full adder can always be used to implement a half adder if the carry-in is tied low). The diagram below illustrates a combinational circuit for performing the 4x4 binary multiplications.

The initial layer of AND gates forms the sixteen partial products that result from ANDing all combinations of the four multiplier bits with the four multiplicand bits. The column sums are formed using a combination of half and full adders.

The adder blocks (indicated by FA and HA) in the figure above are drawn in such a way that the two bits to be added enter from the top, any carry in from the right enters from the right, and any carry out exits from the left of each block. The output from the bottom of a block is the sum.

The least significant output bit, S0 (the first column), involves only two input bits and is computed as the simple output of an AND gate. This operation cannot generate a carry out. The next output bit, S1, involves the sum of two partial products. A half adder is used to form the sum since there can be no carry in from the first column; however, a carry out can be produced. The third output bit, S2, is formed from the sum of three (1-bit) partial products plus a possible carry in from the previous bit. This operation requires two cascaded adders (one half adder and one full adder) to sum the four possible input bits (three partial products and one possible carry in from the right). The remaining output bits are formed similarly. Because in some columns we must add more than two binary numbers, there may be more than one carry out generated to the left.

3.3 MODIFIED BOOTH ALGORITHM

The Modified Booth Multiplier was proposed by O. L. Macsorley in 1961. The recoding method is broadly used to produce the partial products for implementation of large parallel multipliers, which adopts the parallel encoding scheme. One of the solutions of realizing high speed multipliers is to develop parallelism which helps to reduce the number of subsequent stages. The original version of Booth algorithm (Radix-2) had two drawbacks:

- The number of add subtract operations and the number of shift operations becomes variable and becomes inconvenient in designing parallel multipliers.
- The algorithm becomes inefficient when there are isolated 1’s.

These problems can be overcome by Modified Booth algorithm (MBA). In MBA process three bits at a time are recorded. Recoding the multiplier in higher radix is a powerful way to speed up standard Booth multiplication algorithm. In each cycle a greater number of bits can be inspected and eliminated therefore total number of cycles required to obtain products get reduced. Number of bits inspected in radix r is given by n = 1 + log2r.

In each cycle of radix-4 algorithm, 3 bits get inspected. Procedure for implementing radix-4 algorithm is as follows

- Append a 0 to the right of LSB
- Extend the sign bit 1 position if necessary to ensure that n is even
- According to the value of each vector, find each partial product

Radix-4 encoding reduces the total number of multiplier digits by a factor of two, which means in this case the number of multiplier digits will reduce from 16 to 8. Booth's recoding method does not propagate the carry into subsequent stages. This algorithm groups the original multiplier into groups of three consecutive digits where the outermost digit in each group is shared with the outermost digit of the adjacent group. Every one of these group of three binary digits then corresponds to one of the numbers from the set {2, 1 0, - 1,-1}. Each recoder produces a 3-bit output where the first bit represents the number 1 and the second bit represents this number 2. The third and final bit indicates whether the number in the first or second bit is negative. The Modified Booth algorithm is encoded in a table which is shown in table

Table 3.1 Modified Booth Algorithms

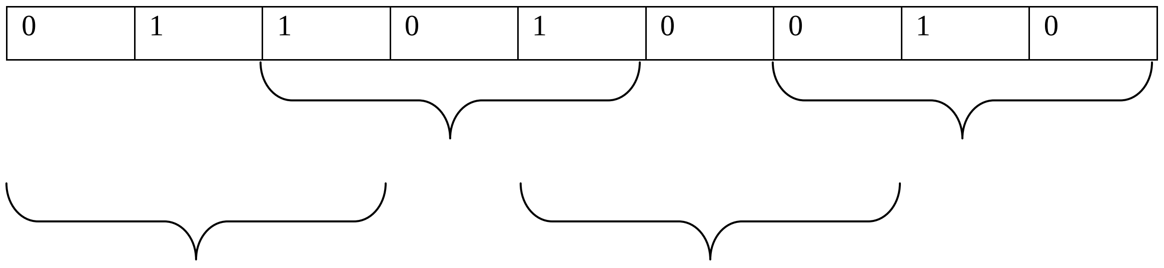
Y_{i+1}	Y_i	Y_{i-1}	Recoded Digit	Operand Multiplication
0	0	0	0	0*Multiplicand
0	0	1	+1	+1*Multiplicand
0	1	0	+1	+1* Multiplicand
0	1	1	+2	+2* Multiplicand
1	0	0	-2	-2* Multiplicand
1	0	1	-1	-1* Multiplicand
1	1	0	-1	-1* Multiplicand
1	1	1	0	0* Multiplicand

Let the two numbers to be multiplied are X= -107and Y=105

Multiplicand X= -107=10010101

Multiplier Y= 105=01101001

X*Y= -11235



0	1	1	0	1	0	0	1	0

Groups			Coding
0	1	0	1xY
1	0	0	-2xY
1	0	1	-1xY
0	1	1	2xY

Table 3.2 Example of Modified Booth algorithm

								1	0	0	1	0	1	0	1	-107
								0	1	1	0	1	0	0	1	105
							1	1	0	0	1	0	1	0	1	PP1
					0	1	1	0	1	0	1	1	0			PP2
			0	0	1	1	0	1	0	1	1					PP3
	1	0	0	1	0	1	0	1	0							PP4
1	1	0	1	0	1	0	0	0	0	0	1	1	1	0	1	-11235

Where PP1, PP2, PP3, PP4 are the partial products formed. The first partial product is determined by three digits LSB of multiplier with an appended zero. This 3 digit number is 010 which mean that the multiplicand X has to multiply by 1. Hence, the first partial product is 110010101. All of the partial products will have nine bits length. Next, the second partial product is computed by next three bits i.e. multiply by -2. Multiply by -2 means the multiplicand value has to shift left one bit and then take two's complement of that number. So, the second partial product is 011010110. Similarly the third partial product has to multiply by -1. Multiply by -1 means the multiplicand has to convert to two's complement value. So, the forth partial product is 100101010. The fourth partial product is determined by next three bits i.e. to multiply by 2.

Multiply by 2 means the multiplicand value has to shift left one bit. So, the forth partial product is 100101010.

LSB of each block gives information about sign bit of the pervious block, and there are never any negative products before the least significant block, so LSB of first block is always taken to be zero. In case where there are not enough bits to obtain the MSB of last block, multiplier is sign extended by one bit. Therefore Modified Booth Multiplication is a technique that allows for smaller, faster circuits by recoding the numbers that are multiplied. It is a standard technique used in chip design, and provides significant improvement over long multiplication technique.

Advantages and applications of modified booth algorithms are described below

Advantages

- Booth multipliers save costs (time and area) for adding partial products
- With the higher radix the number of additions is reduced and the redundant Booth code reduces costs for generating partial products in a higher radix system.
- Low power consumption is achieved in radix 4 booth multiplier because it is a high speed parallel multiplier.

Applications

- Multimedia and communication systems
- Real-time signal processing like audio signal processing, video/image processing, or large capacity data processing.

The multiplier and multiplier-and-accumulator (MAC) are the essential elements of the digital signal processing such as filtering, convolution, and inner products.

3.4 TECHNIQUES FOR IOS

There are three different techniques to decrease this overall power consumption, technique A, B and C each of these algorithms constitutes a different approach to process the mixed-precision data held in the operands buffer.

* The technique A is performed by varying the precision low to high viz... 8bit, 16bit, 32bit.
* The technique B is performed by keeping the voltage constant and varying the frequency of multipliers viz... 8bit, 16bit, 32bit from high to low.
* The technique C is performed by varying the precision high to low viz... 8bit, 16bit, 32bit.

3.5 AMPLITUDE DETECTION LOGIC

The proposed design of adaptive filter is used an amplitude detection logic block, which is placed before the multiplier in Fig 3.12. It is used for selection of multipliers like 8bit, 16bit, 32bit. The following block diagram shows the amplitude detection logic

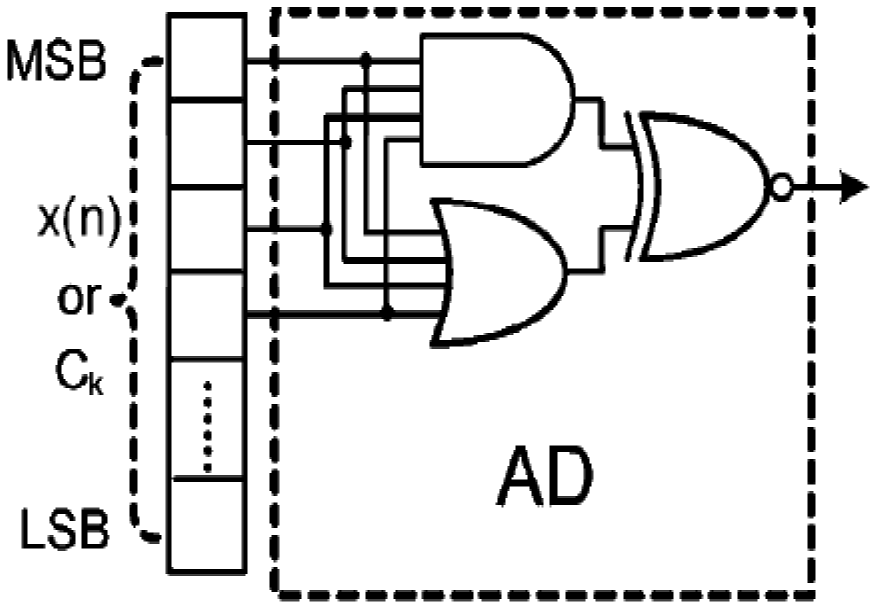


Fig 3.6 Block diagram of Amplitude Detection Logic

3.6 LMS ALGORITHM

The Least Mean Square (LMS) algorithm, introduced by Widrow and Hoff in 1959 is an adaptive algorithm, which uses a gradient-based method of steepest decent. The term “stochastic gradient” is intended to distinguish the LMS algorithm from the method of steepest descent, which uses a deterministic gradient in a recursive computation of the Wiener filter for stochastic inputs. A significant feature of the LMS algorithm is its simplicity. Indeed, it is the simplicity of the LMS algorithm that has made it the standard against which other linear adaptive filtering algorithms are benchmarked.

The LMS algorithm is a linear adaptive filtering algorithm, which, in general, consists of two basic processes:

∗ A filtering process, which involves (a) computing the output of a linear filter in response to an input signal and (b) generating an estimation error by comparing this output with a desired response.
∗ An adaptive process, which involves the automatic adjustment of the parameters of the filter in accordance with the estimation error.

The combination of these two processes working together constitutes a feedback loop, as illustrated in the block diagram of Fig 3.7. First, we have a transversal filter, around which the LMS algorithm is built; this component is responsible for performing the filtering process. Second, we have a mechanism for performing the adaptive control process on the tap weights of the transversal filter- hence the designation “adaptive weight control mechanism” in the figure.

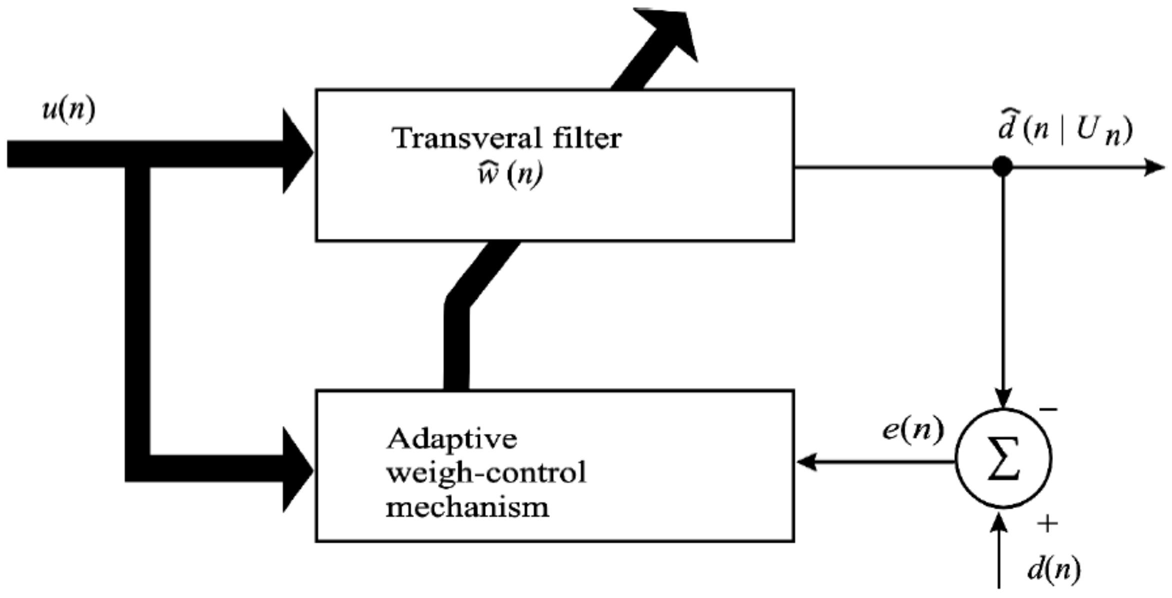


Fig 3.7 Block diagram of adaptive transversal filter

Details of the transversal filter component are presented in Fig 3.8. The tap inputs u(n),u(n-1),…..,u(n-M+1) form the elements of the M-by-1 tap-input vector u(n), where M-1 is the number of delay elements; these inputs span a multidimensional space denoted by u_{n}. Correspondingly, the tap weights $\hat{w}_0\,(n), \hat{w}_1\,(n),......, \hat{w}_{M-1}\,(n)$ form the elements of the M-by-1 tap weight vector $\hat{w}(n)$. The value computed for this vector using the LMS algorithm represents an estimate whose expected value may come close to the Wiener solution w_0 (for a wide sense stationary environment) as the number of iterations, n, approaches infinity.

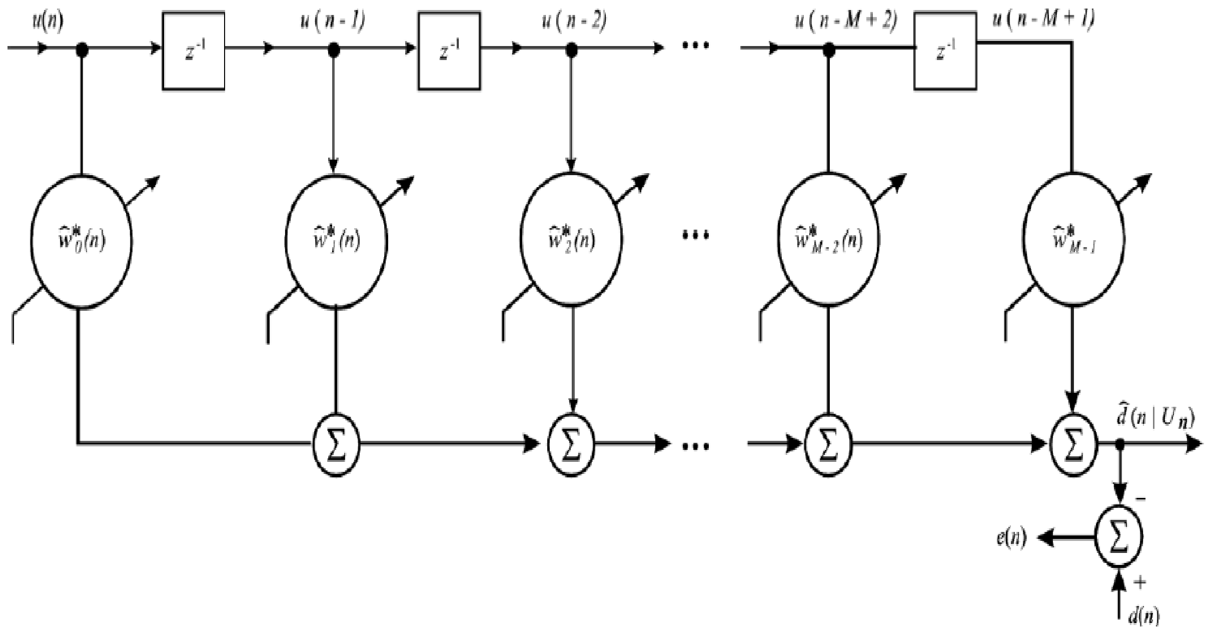


Fig 3.8 Detailed structure of the transversal filter component

During the filtering process, the desired response d (n) is supplied for processing, alongside the tap input vector u (n). Given this input, the transversal filter produces an output $\hat{d}(n|u_n)$ used as an estimate of the desired response d (n). Accordingly we may define an estimated error e (n) as the difference between the desired response and the actual filter output, as indicated in the output end of Fig 3.8. The estimation error e (n) and the tap-input vector u (n) are applied to the control mechanism, and the feedback loop around the tap weights is thereby closed.

Fig 3.9 presents details of the adaptive weight-control mechanism. Specifically, a scalar version of the inner product of the estimation error e(n) and the tap input u(n-k) is computed for k=0,1,2,….,M-2,M-1. The result so obtained defines the correction $\delta \hat{w}_k\ (n)$ applied to the tap weight $\hat{w}_k\ (n)$ at iteration n+1. The scaling factor used in this computation is denoted by a positive quantity µ in Fig 3.9 called the step size parameter.

Comparing the control mechanism of Fig 3.9 for the LMS algorithm with the method of steepest descent, we see that the LMS algorithm uses the product u (n-k) $e^*(k)$ as an estimate of element k in the gradient vector ΔJ (n) that characterizes the method of steepest descent. In other words, the expectation operator is removed from all the paths in Fig 3.9 Accordingly, the recursive computation of each tap weight in the LMS algorithm suffers from a gradient noise.

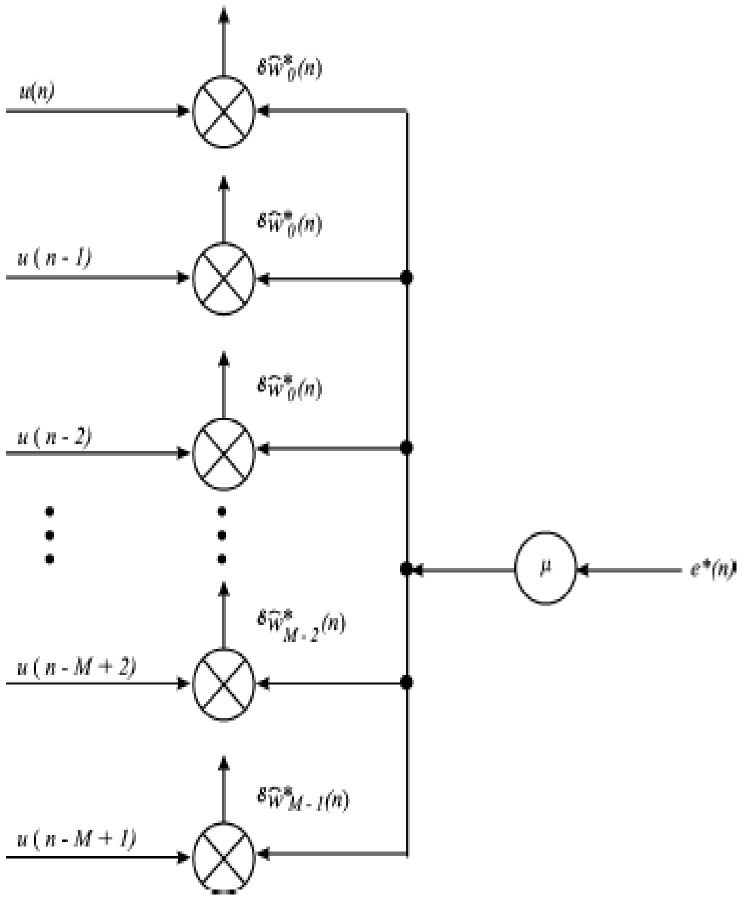


Fig 3.9 Detailed structure of adaptive weight-control mechanism

LMS algorithm uses the estimates of the gradient vector from the available data. LMS incorporates an iterative procedure that makes successive corrections to the weight vector in the direction of the negative of the gradient vector which eventually leads to the minimum mean square error. Compared to other algorithms LMS algorithm is relatively simple.

The LMS algorithm is an efficient and simple adaptive algorithm which mainly employed for adaptive algorithm which mainly employed for adaptive filtering. In this algorithm the filter weights are updated for tracking the desired filter output using the error information. The following mathematical model describes the LMS algorithm.

1. Filter output

$$y(n) = w^T(n)\, x(n)$$

2. Estimation error or error signal

$$e(n) = d(n) - y(n)$$

3. Tap-weight adaptation

$$w(n+1) = w(n) + \mu e(n)\, u(n)$$

where $y(n)$ is the filter output, $w^T(n)$ is the filter weights in transposed form, $x(n)$ is the filter inputs usually measured by sensors, $d(n)$ is the desired filter output, $e(n)$ is the error signal which is used for training the filter weights and μ is the learning factor which is used for controlling the stability and the rate of convergence.

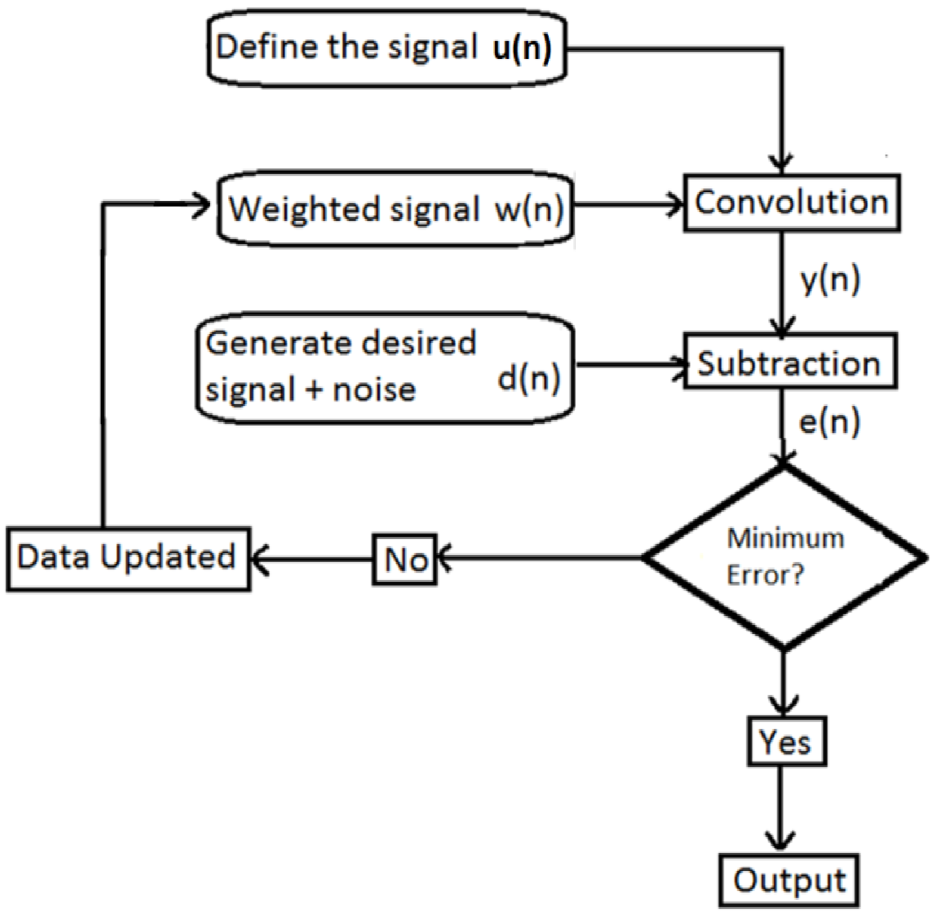


Fig 3.10 Flow chart of LMS algorithm

The LMS algorithm requires only 2M+1 complex multiplications and 2M complex additions per iteration, where M is the number of tap weights used in the adaptive transversal filter. In other words, the computational complexity of the LMS algorithm is O (M).

3.7 ARCHITECTURE OF LMS ADAPTIVE FILTER

Two different commonly used approaches provide the basis of designing FIR filters called direct and transposed architectures. The direct form of an LTI FIR filter is shown in Fig 3.11 which mainly consists of shift registers, adders and multipliers. The signal samples are multiplied by filter coefficients and are gathered together in the adder block. The transposed structure, a modification of direct form, does not consist of shift registers which makes it simpler for implementation than direct form. The direct form of LMS adaptive filter is shown in Fig 3.12

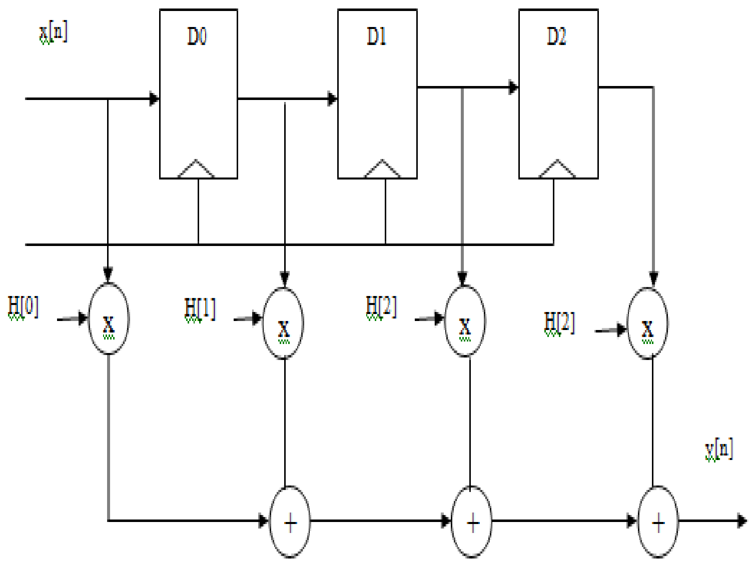


Fig 3.11 Direct form of FIR Filter

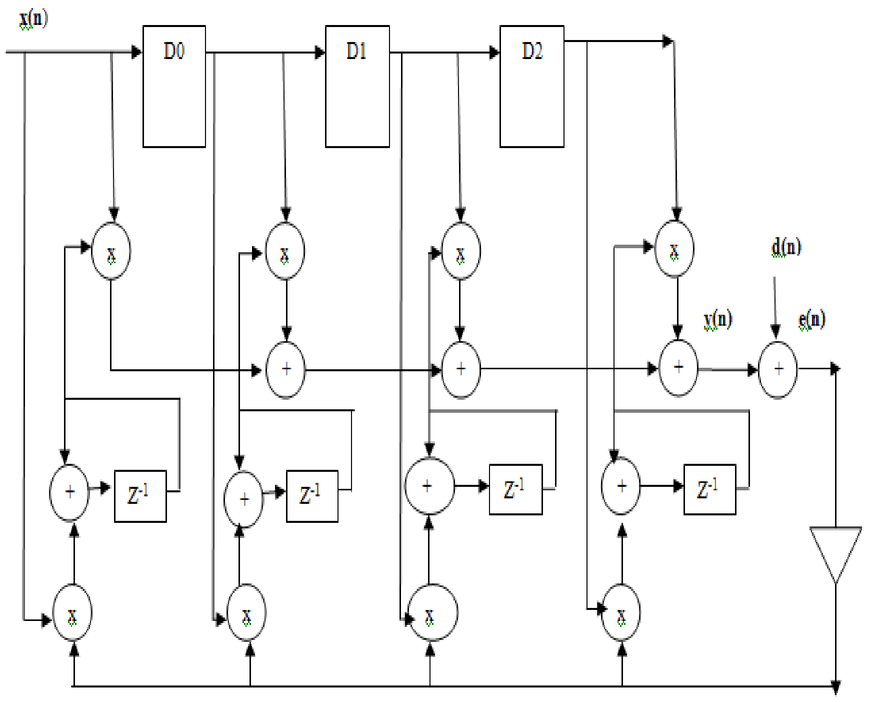


Fig 3.12 Direct form of LMS Adaptive Filter

CHAPTER IV

SIMULATION RESULTS

4.1 INTRODUCTION

The entire module is coded using VHDL and simulated using Xilinx ISE 8.1i software and implemented in Virtex4 DSP family. The proposed design is more efficient in terms of power. This chapter shows the simulation result of the existing design and its power is analyzed with various techniques. This chapter also shows the simulation results of the adaptive filter with various multiplication schemes and its power is compared with adaptive design using Multiprecision multiplier.

4.2 SIMULATION RESULTS OF EXISTING MULTIPRECISION MULTIPLIER

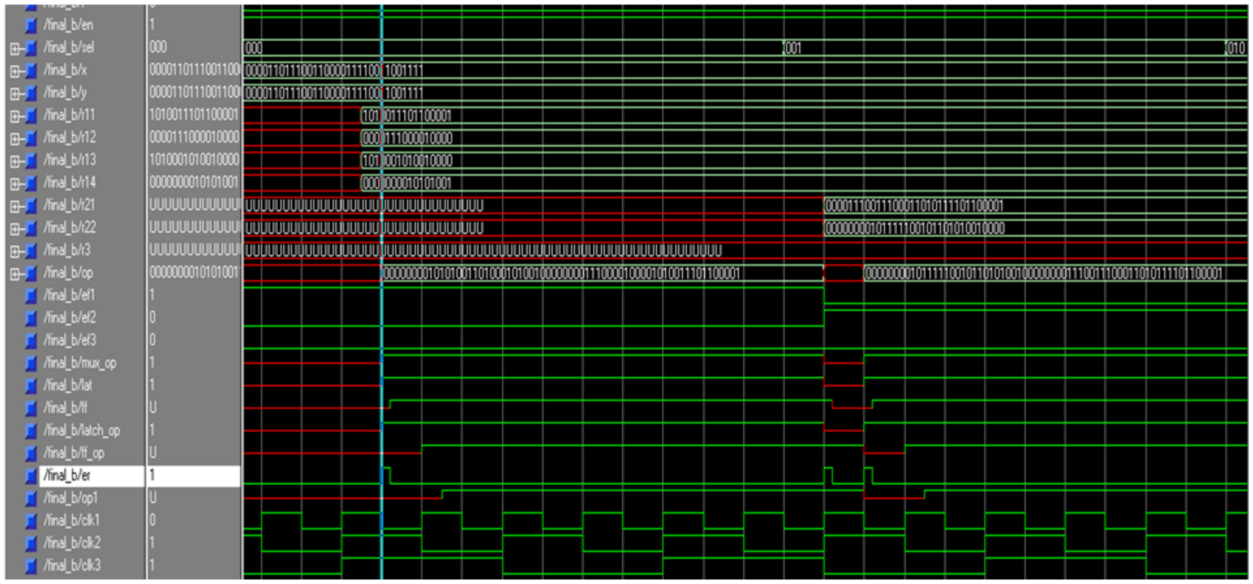


Fig 4.1 Simulation results of Multiprecision Multiplier

This Fig 4.1 shows the simulation results of the Multiprecision multiplier, it also shows the results of changes in the voltages and frequencies whenever the selection lines get changed.

4.3 POWER COMPARISON FOR DIFFERENT TECHNIQUES

Table 4.1 Total power consumption for different techniques at various frequencies

Freq(MHz)	Technique A Power(mW)	Technique B Power(mW)	Technique C Power(mW)
250	355	350	343
500	416	405	392
750	477	460	441
1000	537	515	490

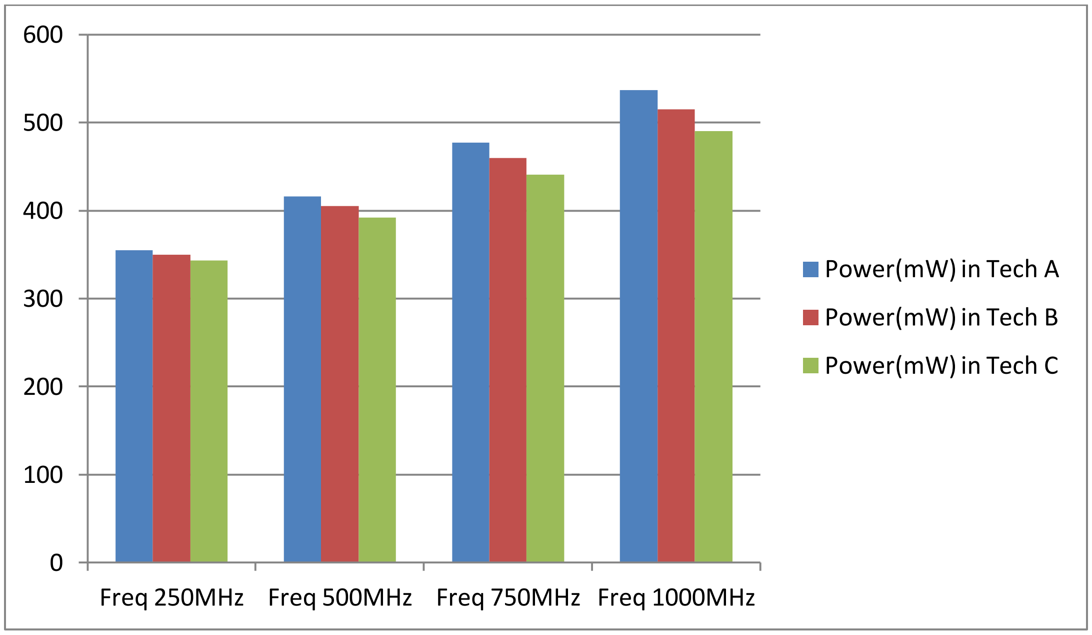


Fig 4.2 Power analysis for various techniques

Table 4.2 shows the comparison of power of different techniques which is described as A, B, C and the Technique C is observed low power compared to technique A and technique B.

4.4 SIMULATION RESULTS OF ADAPTIVE FILTER USING MULTIPRECISION MULTIPLIER

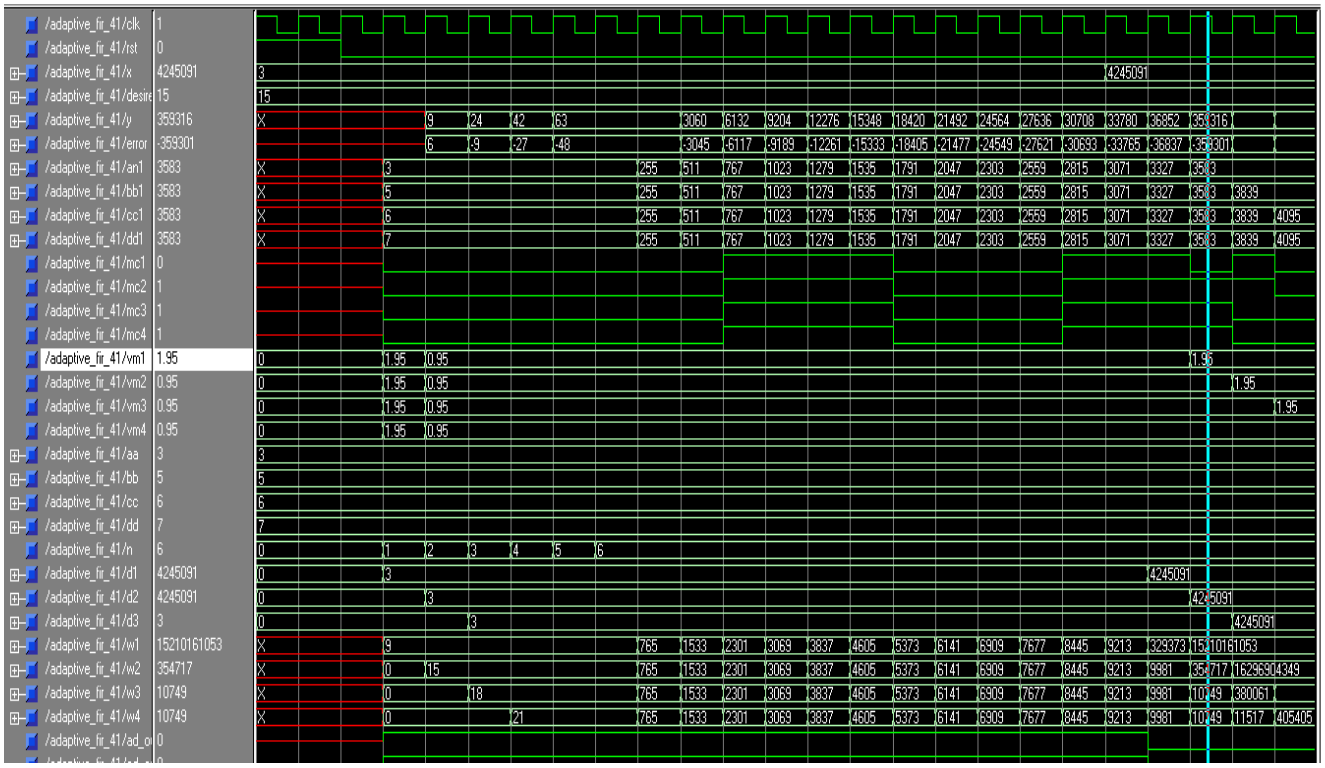
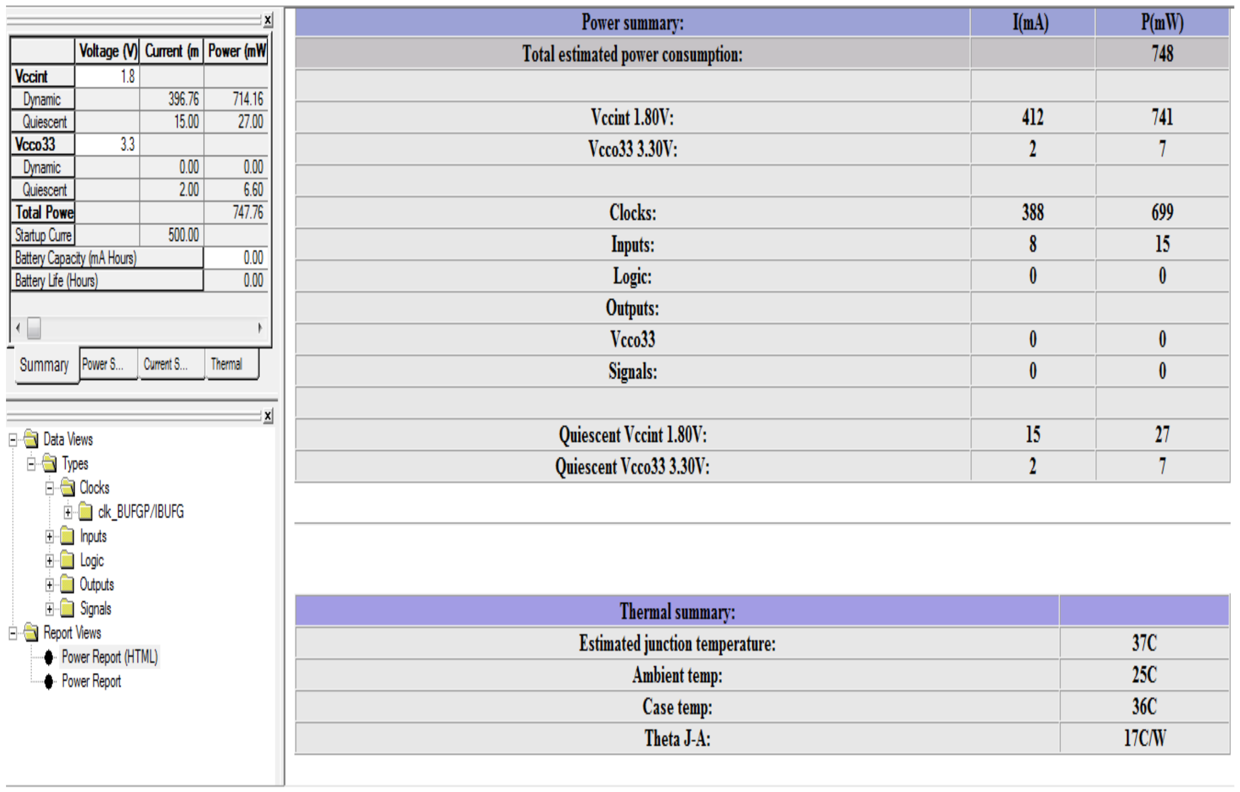

/adaptive_fir_41/clk 1
/adaptive_fir_41/rst 0
/adaptive_fir_41/x 4245091
/adaptive_fir_41/desire 15
/adaptive_fir_41/y 359316
/adaptive_fir_41/error -359301
/adaptive_fir_41/an1 3583
/adaptive_fir_41/bb1 3583
/adaptive_fir_41/cc1 3583
/adaptive_fir_41/dd1 3583
/adaptive_fir_41/mc1 0
/adaptive_fir_41/mc2 1
/adaptive_fir_41/mc3 1
/adaptive_fir_41/mc4 1
/adaptive_fir_41/vm1 1.95
/adaptive_fir_41/vm2 0.95
/adaptive_fir_41/vm3 0.95
/adaptive_fir_41/vm4 0.95
/adaptive_fir_41/aa 3
/adaptive_fir_41/bb 5
/adaptive_fir_41/cc 6
/adaptive_fir_41/dd 7
/adaptive_fir_41/n 6
/adaptive_fir_41/d1 4245091
/adaptive_fir_41/d2 4245091
/adaptive_fir_41/d3 3
/adaptive_fir_41/w1 15210161053
/adaptive_fir_41/w2 354717
/adaptive_fir_41/w3 10749
/adaptive_fir_41/w4 10749
/adaptive_fir_41/ad_o 0

Fig 4.3 Simulation results of adaptive filter using Multiprecision Multiplier

This Fig 4.3 shows the simulations results of adaptive filter design using Multiprecision multiplier, it also shows the results of changing the voltages and frequencies whenever the selection of multiplier like 8bit, 16bit or 32bit get changed

4.5 POWER CONSUMPTION OF ADAPTIVE FILTER USING ARRAY MULTIPLIER



	Voltage (V)	Current (m	Power (mW
Vccint	1.8		
Dynamic		396.76	714.16
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			747.76
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		748
Vccint 1.80V:	412	741
Vcco33 3.30V:	2	7
Clocks:	388	699
Inputs:	8	15
Logic:	0	0
Outputs:		
Vcco33	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

Thermal summary:	
Estimated junction temperature:	37C
Ambient temp:	25C
Case temp:	36C
Theta J-A:	17C/W

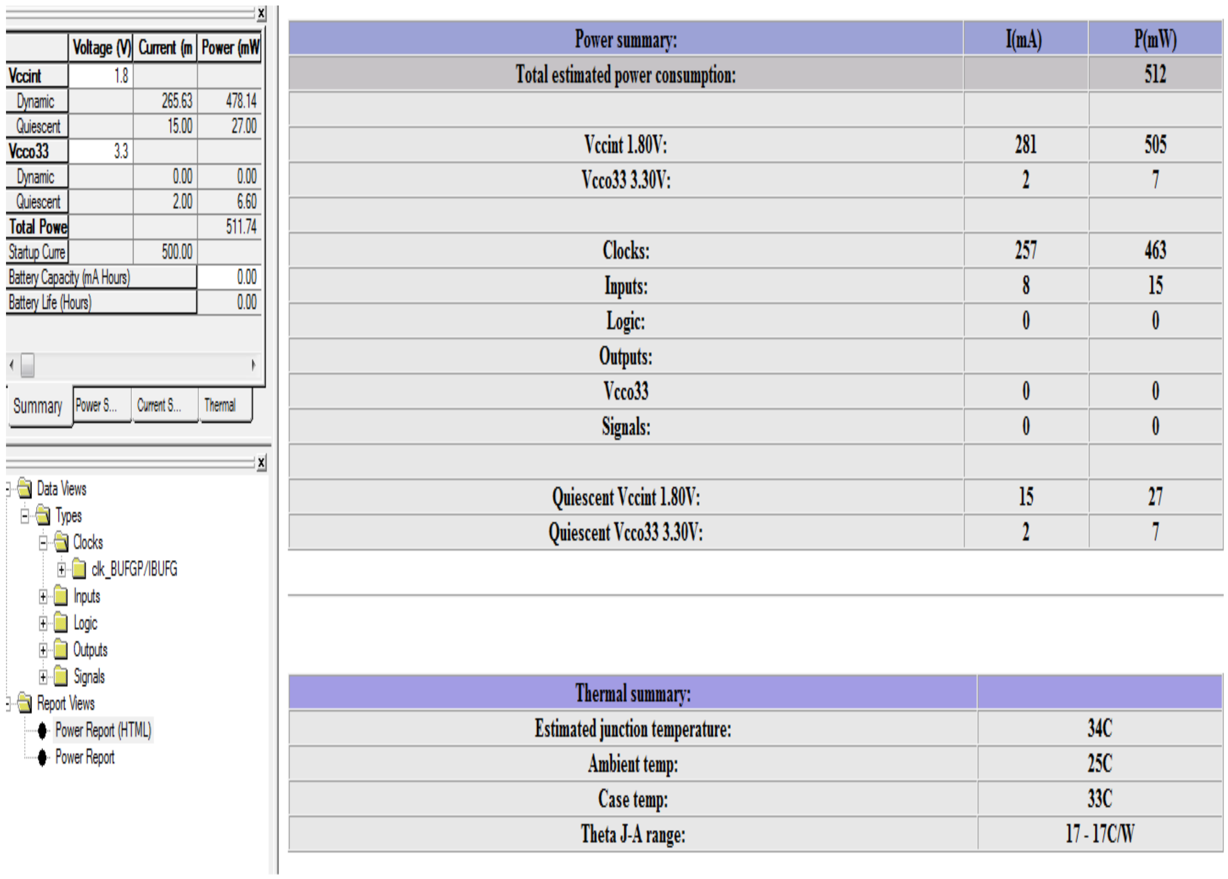
Fig 4.4 Power consumption of the adaptive filter using array multiplier

This Fig 4.4 shows the consumption of the adaptive filter design using array multiplier where frequency is set as 1000MHz.

Specifications are

Family	: Spartan 2E
Device	: XC2S600E
Package	: FG676
Speed	: -7

4.6 POWER CONSUMPTION OF ADAPTIVE FILTER USING MODIFIED BOOTH MULTIPLIER



	Voltage (V)	Current (m	Power (mW
Vccint	1.8		
Dynamic		265.63	478.14
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			511.74
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		512
Vccint 1.80V:	281	505
Vcco33 3.30V:	2	7
Clocks:	257	463
Inputs:	8	15
Logic:	0	0
Outputs:		
Vcco33	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

Thermal summary:	
Estimated junction temperature:	34C
Ambient temp:	25C
Case temp:	33C
Theta J-A range:	17 - 17C/W

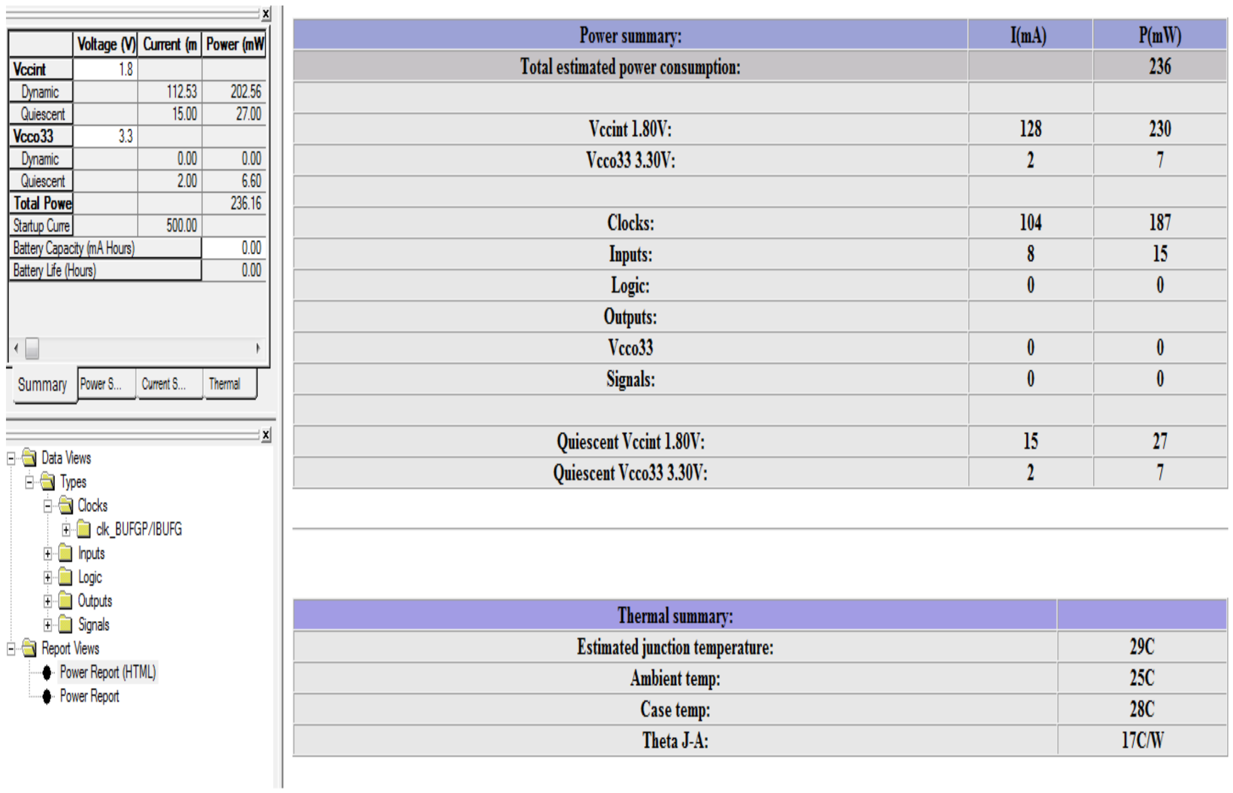
Fig 4.5 Power consumption of the adaptive filter using Modified Booth multiplier

This Fig 4.5 shows the consumption of the adaptive filter design using modified booth multiplier where frequency is set as 1000MHz.

Specifications are

Family	: Spartan 2E
Device	: XC2S600E
Package	: FG676
Speed	: -7

4.7 POWER CONSUMPTION OF ADAPTIVE FILTER USING MULTIPRECISION MULTIPLIER



	Voltage (V)	Current (m	Power (mW
Vccint	1.8		
Dynamic		112.53	202.56
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			236.16
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		236
Vccint 1.80V:	128	230
Vcco33 3.30V:	2	7
Clocks:	104	187
Inputs:	8	15
Logic:	0	0
Outputs:		
Vcco33	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

Thermal summary:	
Estimated junction temperature:	29C
Ambient temp:	25C
Case temp:	28C
Theta J-A:	17C/W

Fig 4.6 Power consumption of the adaptive filter using Multiprecision multiplier

This Fig 4.6 shows the consumption of the adaptive filter design using array multiplier where frequency is set as 1000MHz.

Specifications are

Family	: Spartan 2E
Device	: XC2S600E
Package	: FG676
Speed	: -7

4.8 POWER COMPARISON OF ADAPTIVE FILTER WITH VARIOUS MULTIPLIERS

Table 4.2 Power comparison of adaptive filter with various multipliers

Freq(MHz)	Array Multiplier Power(mW)	Modified Booth Multiplier Power(mW)	Multiprecision Multiplier Power(mW)
250	212	153	84
500	391	273	135
750	569	392	186
1000	748	512	236

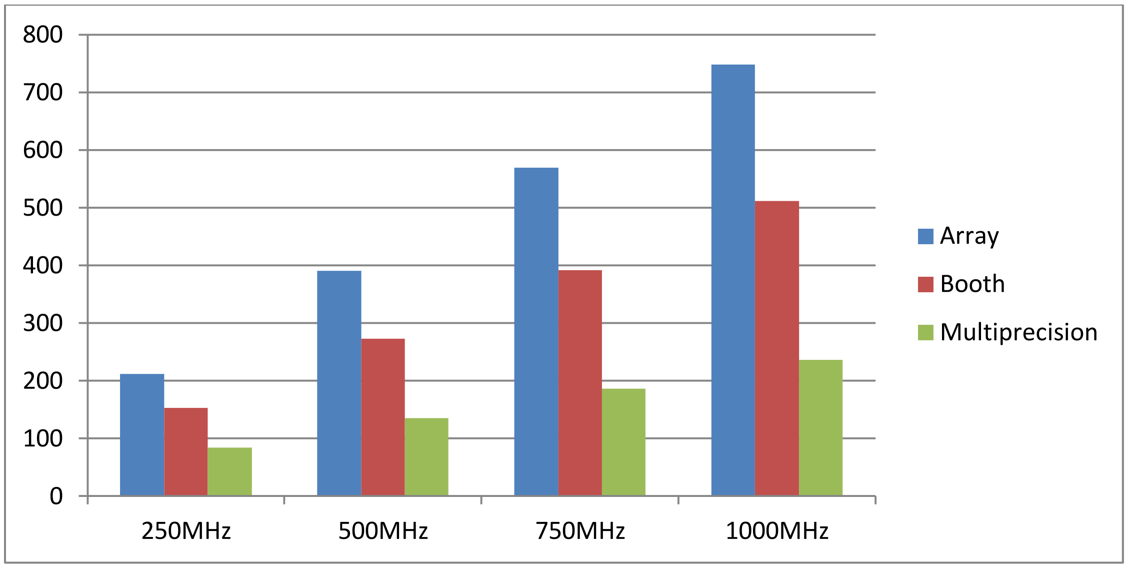


Fig 4.7 Power analysis for adaptive filter using different multiplier at various frequencies

This Fig 4.7 shows the results of power analysis for adaptive filter design using different multiplier at various frequencies

CHAPTER V

CONCLUSION AND FUTURE WORK

In this project, an adaptive filter design using Multiprecision multiplier is designed. It is integrated with razor based dynamic voltage scaling approach with the operand scheduler by which the total power is reduced. Power consumption of this adaptive filter design using Multiprecision multiplier is compared with adaptive filter design using array multiplier and Modified Booth Multiplier. The proposed multiplier architecture for the adaptive filter design is proved to have better power efficiency. The proposed method has 53.91% power reduction than that of the adaptive filter design using modified booth multiplier and 68.45% power reduction than that of the array multiplier. In future work Multiprecision Multiplier can be implemented using FIR, IIR, Decimation filter etc for various applications and performance evaluation for various adaptive filter algorithms can be done.

REFERENCES

[1] A. Y. Lin, K. S. Gugel, and J. C. Principe, “Feasibility of fixed-point transversal adaptive filters in FPGA devices with embedded DSP blocks,” In Proceedings of the 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications 2003, pp. 157–160, 30 June-2 - July 2003.

[2] U. Meyer Baese, “Digital Signal Processing with Field Programmable Gate Arrays”, Springer, 2006.

[3] B. Widrow and S. D. Steams, “Adaptive Signal Processing”. Englewood Cliffs, NJ: Prentice-Hall, 1985.

[4] G. Yecai, H. Longqing, and Z. Yanping, “Design and implementation of adaptive equalizer based on FPGA,” In Proceedings of IEEE 8th International Conference on Electronic Measurement and Instruments 2007 (ICEMI’07), pp. 4-790 – 4-794, August 16 2007-July 18 2007.

[5] M. Vella, and C. J. Debone, “The implementation of a high speed adaptive FIR filter on a field programmable gate array,” In Proceedings of IEEE Electro technical Conference 2006 (MELECON'06), May 16-19, Benalmadena (Malaga), Spain, pp.113–116, 16-19 May 2006.

[6] W. C. Chew, and B. Farhang-Boroujeny, “FPGA implementation of acoustic echo cancelling,” In Proceedings of the IEEE Region 10 Conference 1999 (TENCON’99), Vol. 1, pp. 263–266, 15-17 September 1999.

[7] Hesam Ariyadoost, Yousef S. Kavian, and Karim Ansari-As, “Performance Evaluation of LMS and DLMS Digital Adaptive FIR Filters by Realization on FPGA,” Int. J Sci.Emerging Tech., Vol. 1 No. 1 September, 2011.

[8] Xiaoxiao Zhang, Farid Boussaid, and Amine Bermak, “32 Bit×32 Bit Multiprecision Razor-Based Dynamic Voltage Scaling Multiplier With Operands Scheduler”,IEEE transactions on Very Large Scale Integration (VLSI) systems, vol. 22, no. 4, April 2014.

[9] Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T.Mudge, “Razor: A low-power pipeline based on circuit-level timing speculation,” in *Proc. Int. Symp Microarchit.*, Dec. 2003, pp. 7–18.

[10] Leonardo L.de Oliveira, Eduardo Costa, Sergio Bampi, João Baptista and José Monteiro, “Array Hybrid Multiplier versus Modified Booth Multiplier: Comparing Area and Power Consumption of Layout Implementations of Signed Radix-4 Architectures”, IEEE, 2004.

[11] S. Shafiulla Basha, Syed. Jahangir Badashah, “Design and Implementation of Radix-4 Based High Speed Multiplier for Alu’s Using Minimal Partial Products”, International Journal of Advances in Engineering & Technology, July 2012.

[12] Jayashree Taralabenchi, Kavana Hegde, Soumya Hegde, Siddalingesh S. Navalgund, “Implementation of Binary Multiplication using Booth and Systolic Algorithm on FPGA using VHDL”, International Conference & Workshop on Recent Trends in Technology, (TCET) 2012.

[13] S. A. Shinde, R. K. Kamat, “FPGA based Improved Hardware Implementation of Booth Wallace Multiplier using Handel C”, ISSN 1392 – 1215, 2011.

[14] Y. Sangamitra, “High-Accuracy Fixed-Width Modified Booth multipliers For Lossy Applications”, International Journal of Communications and Engineering, Volume 01–No.1, Issue: 01 March2012.

[15] M. Nakai, S. Akui, et al., "Dynamic voltage and frequency management for a low-power embedded microprocessor," IEEE Journal of Solid-State Circuits, vol. 40, no. 1, Jan. 2005, pp 28- 35.

[16] S. Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull and D. T. Blaauw, “Razor II: In Situ Error Detection and Correction for PVT and SER Tolerance”, Proc. ISSCC, 2008, pp. 400–622.

[17] Seok-Jae Lee, Student Member, IEEE , Ji-Woong Choi , Senior Member, IEEE , Seon Wook Kim , Member, IEEE ,and Jongsun Park, Member, IEEE “A Reconfigurable FIR Filter Architecture to Trade Off Filter Performance for Dynamic Power Consumption” IEEE transactions on very large scale integration (VLSI) systems, vol. 19, no. 12, December 2011

[18] Shweta S. Khobragade, Swapnali P. Karmore, “Review on: Low Power VLSI Design of Modified Booth Multiplier” International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-2, Issue-5, June 2013

[19] M. Bhardwaj, R. Min and A. Chandrakasan, “Quantifying and Enhancing Power-Awareness of VLSI Systems”, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 9, issue 6, Dec. 2001, pp. 757-772.

[20] Chong, Kwen-Siong, Bah-Hwee Gwee, Chang, Joseph.S, “A micro power low-voltage multiplier with reduced spurious switching” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on (Volume:13 , Issue: 2) Feb. 2005

[21] Stefania Perri, Pasquale Corsonello, Maria Antonia Iachino, Marco Lanuzza, and Giuseppe Cocorullo,“Variable precision arithmetic circuits for FPGA-based multimedia Processors” IEEE Trans. VLSI Syst. 12(9):995-999 (2004)

[22] Thomas D. Burd, Trevor A. Pering, Anthony J. Stratakos, Robert W. Brodersen, “A dynamic voltage scaled microprocessor system” IEEE Journal of Solid-state Circuits - IEEE J SOLID-STATE CIRCUITS , vol. 35, no. 11, pp. 1571-1580, 2000

[23] Kavita, Jasbir Kaur, “Design and Implementation of an Efficient Modified Booth Multiplier using VHDL”, International Conference on Emerging Trend in Engineering and Management, ISSN: 2231- 0347, Vol.3 (3, July2013).

[24] Simon Haykin, “Adaptive filter theory”, Third edition, Prentice Hall, 2002.